CARD READER

1. CR11 DIAGNOSTIC
2. PDP-11 DEVICE DRIVER PACKAGE
3. PDP-11 CR 11/CM11 CARD READER DRIVER

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZCRA-A-D
PRODUCT NAME:	CR11 DIAGNOSTIC
DATE:	APRIL 15, 1972
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	PATRICK FADDEN
REPLACES:	MAINDEC-11-D9CB

COPYRIGHT © 1971, 1972
DIGITAL EQUIPMENT CORPORATION

1, ABSTRACT

THIS TEST IS TO BE USED AS A CARD READER DIAGNOSTIC FOR THE PDP-11 WITH THE CR11 CARD READER. IT TESTS ALL LOGIC FUNCTIONS OF THE CARD READER, AND INCLUDES AN EXERCISER FOR ALPHANUMERIC AND BINARY TEST DECKS. A SEPARATE STARTING ADDRESS ALLOWS THE ERROR SENSING FUNCTIONS OF THE G,D,I, OR DOCUMATION READER TO BE CHECKED. ANOTHER STARTING ADDRESS TESTS SPECIAL DECKS WHICH HAVE ALL COLUMNS AND CARDS PUNCHED IDENTICALLY, TO AID IN DIAGNOSING SPECIAL PROBLEMS.

2, REQUIREMENTS

2,1 EQUIPMENT

PDP-11/20 STANDARD COMPUTER
CR11 CARD READER

2,2 TEST DECKS

MAINDEC-89-D2A1-C ALPHANUMERIC TEST DECK
MAINDEC-89-D2A2-C BINARY TEST DECK
EXTRA CARDS (FOR ERROR FUNCTION TEST)

2,3 STORAGE

THE ROUTINE USES MEMORY 0 TO 15000.

3, LOADING PROCEDURE

PROCEDURE FOR NORMAL ABSOLUTE TAPES SHOULD BE FOLLOWED.

4, STARTING PROCEDURE

4,1 CONTROL SWITCH SETTINGS

BASIC SWITCH REGISTER SETTINGS ARE:

SW15=1 OR UP---HALT ON ERROR
SW14=1 OR UP---SCOPE LOOP
SW13=1 OR UP---INHIBIT PRINT OUT
SW12=1 OR UP---INHIBIT TRACE TRAPPING
SW11=1 OR UP---INHIBIT SUB-PROGRAM ITERATION
(NOTE THAT IF SW11 IS SET, THE CARD COUNT WILL BE ALTERED, CAUSING FAILURES IN THE DATA TEST SECTION.)
SW10=1 OR UP---CR11 CONTROLLER USES THE M829 MODULE
(IF DOWN, ASSUMES THE M8290 MODULE)
SW07=1 OR UP---LOOP THRU THE INSTRUCTION TEST PORTION
NOTE: DATA ERRORS MAY OCCUR IF SW7 IS SET, THEN CLEARED, ALSO THE TEST MAY HANG WHEN THE INPUT HOPPER GOES EMPTY IF SW7 WAS SET.
SW06=1 OR UP---RETURN TO THE BEGINNING OF THE INSTRUCTION TEST WHEN CONTINUING FROM ONE DECK TO ANOTHER
SW05=1 OR UP---HALT BETWEEN TEST DECKS
(SEE 5.2.1 FOR EXPLANATION OF SW5=0)
SW04=1 OR UP---RUN THE BINARY TEST DECK

4,2 STARTING ADDRESSES

200 = INSTRUCTION AND DATA TEST
210 = ERROR FUNCTION TEST (WITH G,D,I, READER)
220 = ERROR FUNCTION TEST (WITH DOCUMATION READER)
240 = SINGLE SUBTEST LOOP
250 = READ SINGLE DATA PATTERN TEST

4,3 PROGRAM AND/OR OPERATOR ACTION

4,3,1 INSTRUCTION AND DATA RELIABILITY TEST (SA 200)

LOAD PROGRAM INTO MEMORY,
LOAD ONE TEST DECK IN THE CARD READER INPUT HOPPER,
PRESS MOTOR START AND READ START ("RESET" ON DOCUMATION READER),
SET SWITCH REGISTER TO STARTING ADDRESS,
LOAD ADDRESS,
SET SWITCHES (SEE 4,1)--ALL DOWN FOR WORST CASE,
PRESS START,
WHEN THE INPUT HOPPER IS EMPTY THE PROGRAM WILL HANG WAITING
FOR AN INTERRUPT FROM THE CARD READER, LOAD ONE OR MORE
TEST DECKS INTO THE INPUT HOPPER, PRESSING "MOTOR START" AND
"READ START" ("RESET" ON DOCUMATION READER) ON THE CARD
READER SHOULD CAUSE PROGRAM EXECUTION TO RESUME,
THIS ENTIRE SEQUENCE IS NECESSARY TO RUN THE FULL TEST ON THE CARD
READER,
ALL PRINTOUTS INDICATE FAILURE, INCLUDING THOSE SAYING
THAT BIT 8 OR BIT 15 WAS SET,

4,3,2 ERROR FUNCTION TEST (SA 210 OR 220)

LOAD A FEW SPARE CARDS INTO THE INPUT HOPPER (DO NOT LOAD A
TEST DECK-THIS TEST IS DESTRUCTIVE!)
PRESS "MOTOR START" AND "READ START" ("RESET" ON DOCUMATION READER)
ON THE CARD READER,
LOAD THE STARTING ADDRESS, THEN SET THE DESIRED
SWITCH OPTIONS,
PRESS START,
FOLLOW THE INSTRUCTIONS AS THEY ARE PRINTED OUT,

4,3,3 SINGLE SUBTEST LOOP (SA 240)

LOAD CARDS (SPARE CARDS OR A TEST DECK) INTO THE INPUT HOPPER,
PRESS "MOTOR START" AND "READ START" ("RESET" ON DOCUMATION READER)
ON THE CARD READER,
LOAD THE STARTING ADDRESS,
PRESS START,
AT THE 1ST HALT, LOAD THE STARTING ADDRESS OF THE DESIRED TEST
(ADDRESS OF THE TESTXX TAG, WHERE XX MAY BE 1 THRU 24
OR A THRU G),
PRESS CONTINUE,
AT THE 2ND HALT SET THE SWITCH REGISTER OPTIONS (BIT 11 MUST=0),
PRESS CONTINUE,

4,3,4 SINGLE DATA PATTERN TEST (SA 250)

A SPECIAL DECK (1 OR MORE CARDS) MUST BE PUNCHED TO RUN THIS TEST.
ANY DATA PATTERN MAY BE USED, BUT IT MUST BE IDENTICAL IN ALL
80 COLUMNS OF ALL THE CARDS (I.E. ONLY ONE PIECE OF DATA).
LOAD THIS PREPARED DECK INTO THE INPUT HOPPER.
PRESS CARD READER "MOTOR START" AND "READ START" ("RESET" ON
DOCUMATION READER).
LOAD SA 250.
PRESS START.
AT THE INITIAL HALT SET THE CARD IMAGE OF THE DATA PATTERN USED
IN SW11-SW00.
PRESS CONTINUE.
WHEN THE CARD READER RUNS OUT OF CARDS IT WILL RING THE BELL.
RELOADING THE DECK AND PRESSING "READ START" ("RESET") ON THE CARD
READER WILL CONTINUE THE TEST.

5, OPERATING PROCEDURE

5,1 OPERATIONAL SWITCH SETTINGS

5,1,1 AT SA 200 (INSTRUCTION AND DATA RELIABILITY TEST)

SEE 4,1

5,1,2 AT SA 210 OR 220 (ERROR FUNCTION TEST FOR CR11)

SW00=1 TO INHIBIT TESTING THE DARK-LIGHT ERROR.
SW14=1 TO LOOP THRU THE CURRENT SUBTEST
SW15=1 TO HALT ON ERROR

5,1,3 AT SA 240 (SINGLE SUBTEST LOOP)

1ST HALT = LOAD STARTING ADDRESS OF DESIRED TEST
2ND HALT = SET SR OPTIONS (BIT 11 MUST=0)
SEE 4,1 FOR SR OPTIONS

5,1,4 AT SA 250 (SINGLE DATA PATTERN TEST)

AT THE HALT-LOAD THE CARD-IMAGE OF THE DATA PATTERN IN SW11-SW00.

5,2 SUBROUTINE ABSTRACTS

5,2,1 BEGIN SA 200

THE INSTRUCTION TESTS ARE RUN FIRST, FOLLOWED BY THE DATA RELIABILITY TESTS ON THE REMAINING CARDS IN THE FIRST TEST DECK. AT THE END OF THE DECK THE BELL WILL RING, AND IF SW5=1 THE PROGRAM HALTS. IF SW5=0, PROGRAM ACTION DEPENDS ON THE NUMBER OF TEST DECKS LOADED. IF THERE ARE STILL CARDS IN THE INPUT HOPPER, THE PROGRAM WILL RUN THE DATA RELIABILITY TEST ON THE ENTIRE NEXT DECK. IF THE INPUT HOPPER IS EMPTY AT THE END OF A DECK, THE PROGRAM WILL RUN A SET OF TESTS OF OFF-LINE OPERATIONS. AT THE END OF THESE TESTS, IT WAITS FOR THE CARD READER TO BE PUT BACK ON-LINE. FURTHER CHECKS ARE MADE OF THE OFF-LINE TO ON-LINE OPERATIONS, AND THEN THE DATA RELIABILITY TEST IS RUN ON THE ENTIRE DECK. IF SW5=1, HITTING CONTINUE WILL RESUME PROGRAM OPERATION AFTER THE HALT. IF ALL OTHER SWITCHES WERE DOWN, FOR EXAMPLE, THE DATA RELIABILITY TEST WOULD THEN BE RUN ON THE NEXT DECK. THE OTHER SWITCHES AFFECT PROGRAM FLOW AS NOTED IN 4,1.

5,2,2 SCOPE

THIS SUBROUTINE CALL IS PLACED BETWEEN EACH SUBTEST IN THE INSTRUCTION SECTION. IT RECORDS THE STARTING ADDRESS OF EACH SUB-TEST AS IT IS BEING ENTERED. IF A SCOPE LOOP IS REQUESTED, IT WILL JUMP TO THE START OF THE SUBTEST THAT THE SCOPE LOOP IS REQUESTED FOR. IF SCOPE LOOP IS NOT REQUESTED, THERE WILL BE 1 ITERATION ON THAT SUBTEST BEFORE THE NEXT SUBTEST IS ENTERED. SWITCH 11 ON A 1 INHIBITS ITERATION OF SUBTESTS.

5,2,3 HLT

THIS SUBROUTINE PRINTS OUT THE LOCATION COUNTER AT THE TIME OF FAILURE, AND THE CONTENTS OF THE PROCESSOR STATUS REGISTER. NOTE THAT THE LOCATION COUNTER WILL BE THE ADDRESS OF THE HLT PLUS TWO.

5,2,4 TTRAP

THIS ROUTINE ALLOWS THE TRACE BIT TO BE SET AFTER THE FIRST LOOP OF THE PROGRAM, THE TRACE BIT WILL BE SET ON ALTERNATE LOOPS OF THE INSTRUCTION TEST, AND ON ALL LOOPS OF THE CHANNEL TEST UNLESS SW12 IS SET, THE FIRST INSTRUCTION EXECUTED UPON TRAPPING IS AN "RTI" WHICH RETURNS TO THE INTERRUPTED SEQUENCE, THIS CONTINUES UNTIL THE END OF THE PROGRAM LOOP IS REACHED.

5,2,5 TRAPCATCHER

THIS IS A SERIES OF INSTRUCTIONS STARTING AT LOCATION Ø DESIGNED TO DETECT AND ISOLATE UNEXPECTED TRAPS AND INTERRUPTS TO THE TRAP AND INTERRUPT VECTOR AREA OF MEMORY.

EACH VECTOR ENTRANCE ADDRESS IS LOADED WITH THE ADDRESS OF THE NEXT LOCATION, THE NEXT LOCATION IS LOADED WITH A HALT (ØØØØØØ). THUS AN ILLEGAL TRAP OR INTERRUPT WILL CAUSE A HALT AT THE TRAP LOCATION PLUS TWO.

IF A HALT OCCURS IN THE TRAP OR INTERRUPT AREA, EXAMINE REGISTER SIX, IT WILL CONTAIN THE CURRENT STACK ADDRESS, THE CONTENTS OF THE CURRENT STACK ADDRESS IS THE VALUE OF THE LOACATION COUNTER WHEN THE TRAP OR INTERRUPT OCCURRED.

5,2,6 ERCR11 (ERROR FUNCTION TEST)

THIS TEST CHECKS OPERATION OF THE VARIOUS ERROR SENSING FEATURES OF THE G.D.I. OR THE DOCUMATION CARD READER, CARD READER OFF-LINE, INPUT HOPPER EMPTY, OUTPUT STACKER FULL, FEED ERROR, MOTION ERROR, STACK FAIL, AND DARK-LIGHT ERROR ARE ALL CHECKED.

5,2,7 TESTX (SINGLE TEST LOOP)

THIS ROUTINE ALLOWS A SINGLE SUBTEST TO BE RUN CONTINUOUSLY FOR SCOPE LOOP PURPOSES, WHILE A SCOPE LOOP SWITCH OPTION EXISTS, IT REQUIRES THAT YOU ARE WITHIN THE TEST IN WHICH YOU WISH TO LOOP, IN SOME CASES (SUCH AS WITH INTERMITTENT FAILURES) THAT'S NOT EASY TO DO, THIS SUBROUTINE ALLOWS YOU TO LOAD THE ADDRESS OF ANY TEST FROM TESTØ THRU TEST24 AND TESTA THRU TESTG AT THE HALT AND THEN GO DIRECTLY TO THAT TEST.

5,2,8 CKSAME (SINGLE DATA PATTERN TEST)

THIS TEST IS DESIGNED TO AID IN THE DIAGNOSIS OF DIFFICULT DATA ERROR PROBLEMS AND FACILITATE SOME CARD READER ADJUSTMENTS, IT CONTINUOUSLY READS CARDS WHICH HAVE ALL COLUMNS PUNCHED IDENTICALLY (AND ALL CARDS MUST BE IDENTICAL), CHECKING THE DATA AGAINST A PATTERN SET UP ON THE SWITCHES INITIALLY, ANY ERRORS ARE PRINTED OUT, ALONG WITH A COUNT OF THE TOTAL NUMBER OF CARDS READ AND THE TOTAL NUMBER OF DATA ERRORS WHICH HAVE OCCURRED SINCE THE TEST WAS STARTED.

5,3 PROGRAM AND/OR OPERATOR ACTION

5,3,1 LOADING AND STARTING AT 200 WITH ALL SWITCHES DOWN IS WORST CASE TESTING. A SINGLE ALPHANUMERIC DECK SHOULD BE RUN. THIS EXECUTES AN INSTRUCTION TEST FOLLOWED BY A DATA RELIABILITY TEST. AT THE END OF THE DECK CHECKS ARE MADE OF THE FLAG SETTINGS WHICH SHOULD BE AFFECTED, AND THE PROGRAM WAITS FOR AN INTERRUPT FROM THE READER COMING BACK ON-LINE. AT THE END OF THE FIRST DECK THE OPERATOR SHOULD LOAD ONE OR MORE DECKS IN THE INPUT HOPPER AND PRESS MOTOR START AND READ START (RESET ON THE DOCUMATION READER). IF THE CARD READER IS WORKING PROPERLY, THE BELL WILL RING ONCE WHEN READ START IS PRESSED AND THE ENTIRE DECK WILL BE RUN THRU THE DATA RELIABILITY PORTION OF THE TEST. IF, AFTER READING 80 CARDS, THE INPUT HOPPER IS NOT EMPTY, THE PROGRAM WILL CONTINUE TO THE NEXT DECK. SWITCH OPTIONS MAY BE USED TO ALTER THIS FLOW AS NOTED IN SECTION 4,1.

5,3,2 TO GO DIRECTLY TO A SINGLE SUBTEST AND RUN IT CONTINUOUSLY, USE SA 240. AT THE FIRST HALT, SET THE SWITCH REGISTER TO THE STARTING ADDRESS OF THE DESIRED SUBTEST (I.E. THE ADDRESS OF THE CORRESPONDING TESTXX TAG), AND CONTINUE. AT THE SECOND HALT, SET THE DESIRED SWITCH REGISTER OPTIONS AND CONTINUE (SW11 MUST BE = 0). THE PROGRAM WILL CONTINUOUSLY LOOP THRU THE DESIRED SUBTEST.

6, ERRORS

6,1 ERROR PRINTOUT

6,1,1 STANDARD PRINTOUT

PRINTOUTS ARE IN A TWO-WORD FORMAT. THE FIRST IS THE PC+2 OF THE DETECTED ERROR. THE SECOND IS THE CONTENTS OF THE PROCESSOR STATUS REGISTER WHEN THE ERROR WAS DETECTED.

6.1.2 DATA ERROR PRINTOUT

THE HEADING IS PRINTED OUT ONCE PER TEST DECK, THE COLUMNS HAVE THE FOLLOWING SIGNIFICANCE:

DECK -EITHER ALPHANUMERIC OR BINARY , DEPENDING ON SW4
CARD -THE CARD NUMBER WHERE THE FAILURE OCCURRED
COLUMN -THE COLUMN NUMBER WHERE THE FAILURE OCCURRED
PATTERN -THE CORRECT CARD IMAGE DATA THAT SHOULD HAVE BEEN READ
READ1 -THE CARD IMAGE DATA IS READ TWICE, THIS IS WHAT WAS READ THE FIRST TIME FROM CRB1
READ2 -THIS IS WHAT WAS IN CRB1 AFTER A BRIEF TIMING LOOP. IT SHOULD BE THE SAME AS THE PREVIOUS READING.
CODED -THIS IS WHAT THE DATA SHOULD BE IN ENCODED FORM
READ -THIS IS WHAT WAS READ BY ADDRESSING THE ENCODED BUFFER

DATA ERRORS NOT TRACED TO CARD READER HARDWARE INCLUDE:
A. SW04 NOT SET TO TYPE OF DECK USED
B. CARD MISSING
C. CARD DECK OUT OF PROPER SEQUENCE
D. DAMAGED CARD

6.1.3 SINGLE DATA PATTERN PRINTOUT

THE SINGLE DATA PATTERN TEST PRINTS OUT A HEADING WITH EACH ERROR PRINTOUT. THE COLUMNS HAVE THE FOLLOWING SIGNIFICANCE:

COLUMN -THE COLUMN NUMBER WHERE THE FAILURE OCCURRED.
READ1 -DATA IS READ TWICE, THIS IS THE FIRST READING.
READ2 -THIS IS WHAT WAS READ THE SECOND TIME.
CARDS -THE TOTAL NUMBER OF CARDS (IN OCTAL) THAT HAVE BEEN RUN SINCE THE TEST WAS STARTED.
ERRORS -THE TOTAL NUMBER OF ERRORS DETECTED (IN OCTAL) SINCE THE TEST WAS STARTED.

6.1.4 "BIT 8 WAS SET"

AT THE BEGINNING OF MOST SUBTESTS, BIT 8 (OFF-LINE) IS CHECKED TO MAKE SURE THAT THE READER IS NOT OFF-LINE, IT IS ALSO CHECKED IN THE DATA TEST WHEN AN INTERRUPT OCCURS DUE TO BIT 15 BEING SET. IF BIT 8 IS SET WHEN IT WAS NOT SUPPOSED TO BE, THE ERROR MESSAGE "BIT 8 WAS SET, REMEDY THE ERROR CONDITION AND PRESS 'CONTINUE'," IS PRINTED OUT, THE PROCESSOR THEN HALTS, SINCE THE CARD READER GOES OFF-LINE WHEN A CARD READER FUNCTION ERROR OCCURS (CARD JAM, PICK FAIL, ETC.), THE CARD READER ERROR MUST BE FIXED AND THE READER MUST BE PUT BACK ON-LINE BEFORE THE PROGRAM CAN BE CONTINUED.

6.2 ERROR RECOVERY

IN GENERAL, TEST FAILURES WILL PRINTOUT AN ERROR MESSAGE AND CONTINUE. IF THE "HALT ON ERROR" SWITCH IS SET, HITTING CONTINUE WILL RECOVER. IF THE PROGRAM HANGS UP IN A LOOP, THE ERROR IS LIKELY TO BE A SIGNAL WHICH WAS NEVER RECEIVED. IF A HALT OCCURS IN THE TRAP AND VECTOR AREA THE PROGRAM MUST BE RESTARTED. IF THE PROGRAM HALTS IN THE MAIN FLOW, CONSULT THE LISTING IF NO MESSAGE IS TYPED OUT.

7, RESTRICTIONS

7,1 STARTING PROCEDURE

NONE

7,2 OPERATIONAL RESTRICTIONS

7,2,1 COMBINED INSTRUCTION AND DATA RELIABILITY TEST (SA200)

IF A STANDARD TEST DECK IS NOT BEING USED, SW7 MUST BE SET TO INHIBIT RUNNING THE DATA RELIABILITY PORTION OF THE TEST.

WHEN USING THE STANDARD TEST DECKS, THEY MUST BE IN PROPER SEQUENCE AND IN GOOD CONDITION. IT IS A GOOD IDEA TO NUMBER THE CARDS IN EACH DECK AS SOON AS THE DECK IS RECEIVED.

IF THE CR11 USES AN M829 MODULE SW10 MUST BE SET IN THE SWITCH REGISTER.

7,2,2 ERROR FUNCTION TEST (SA 210 FOR G.D.I. READER - SA 220 FOR DOCUMATION READER)

THE ERROR FUNCTION TEST REQUIRES SPARE CARDS, AS IT BENDS SEVERAL. ALSO, TO RUN THE DARK-LIGHT CHECK 2 CARDS MUST BE SPECIALLY PREPARED. THE TEST WILL TYPE OUT A REQUEST FOR THESE CARDS WHEN THEY ARE NEEDED. TO MAKE THEM:

1. TEAR A SMALL PIECE FROM THE LEADING EDGE OF ONE CARD.
2. TAPE 2 OTHER CARDS TOGETHER TO MAKE ONE "LONG" CARD - IT ONLY NEEDS TO BE ABOUT 1/2 INCH LONGER THAN A REGULAR CARD

7,2,3 SINGLE DATA PATTERN TEST (SA 250) .

A SPECIAL DECK (ONE OR MORE CARDS) MUST BE PREPARED. ALL COLUMNS OF ALL CARDS ARE PUNCHED IDENTICALLY, USING A DATA PATTERN WHICH WILL TEST THE PROBLEM BEING DIAGNOSED.

8, MISCELLANEOUS

8,1 EXECUTION TIME

NOT APPLICABLE

8,2 CARD DECK DESCRIPTION

8,2,1 ALPHANUMERIC

REFERENCE THE ALPHANUMERIC TABLE BEGINNING AT THE TAG ALPCD IN THE LISTING FOR THE CODES PUNCHED FOR EACH OF 80 COLUMNS OF THE FIRST CARD. THE FIRST VALUE GIVEN FOR A COLUMN IS THE CARD IMAGE OF THAT COLUMN, WHILE THE SECOND VALUE IS THE ENCODED FORM OF THE SAME PATTERN. EACH SUCCESSIVE CARD IN THE DECK USES THE SAME SEQUENCE OF CODES ROTATED ONE COLUMN TO THE LEFT.

8,2,2 BINARY

REFERENCE THE BINARY DATA TABLE BEGINNING AT THE TAG BINCD IN THE LISTING FOR THE CODES PUNCHED FOR EACH OF THE 80 COLUMNS OF THE 1ST CARD. AS WITH THE ALPHANUMERIC DECK EACH SUCCESSIVE CARD HAS THE SAME SEQUENCE OF CODES ROTATED ONE COLUMN TO THE LEFT.

8,3 SPECIAL NOTES

IF THE CARD READER GOES OFF-LINE BEFORE THE END OF A CARD, BUSY REMAINS SET UNTIL THE CARD ACTUALLY CLEARS THE READER.

THE CARD READER GOES OFF-LINE DUE TO "INPUT HOPPER EMPTY" AFTER THE 80TH COLUMN OF THE LAST CARD IS READ, BUT BEFORE CARD DONE OCCURS. THUS, THE SPECIAL CONDITION BIT IN THE CSR WILL BE SET BEFORE CARD DONE ON THE LAST CARD.

IF THE CARD READER USES AN M829 MODULE, SW10 MUST BE SET IN THE SWITCH REGISTER. WITH THE M829 MODULE, CARD DONE NEVER OCCURS AFTER THE LAST CARD IN THE INPUT HOPPER IS READ. IF THE CARD READER USES AN M8290 MODULE, SW10 MUST NOT BE SET. WITH THE M8290 MODULE, CARD DONE IS ISSUED AFTER THE LAST CARD IN THE INPUT HOPPER IS READ.

IF BIT 0 OF THE CRS IS CLEARED IMMEDIATELY AFTER BEING SET, THE READING OF A CARD MAY NOT OCCUR. SINCE THIS BIT IS WRITE ONLY, A BIS OR BIC DONE AFTER SETTING BIT 0 MAY CLEAR THE BIT AND PREVENT THE READ FROM OCCURRING.

8.4 TESTING CR11'S WITH NON-STANDARD ADDRESSES

BY SUBSTITUTING INTO THE LOCATIONS KCRS, KCRB1, AND CRB2 THE ADDRESSES OF THE CRS, CRB1, AND CRB2 OF A CARD READER ASSIGNED A NON-STANDARD ADDRESS, AND SUBSTITUTING ITS INTERRUPT VECTOR ADDRESS INTO ADINT, A CR11 MAY BE TESTED AT ANY ADDRESS ASSIGNED TO IT.

9. PROGRAM DESCRIPTION

THIS SET OF TESTS IS DESIGNED TO CHECK ALL OPERATIONS OF THE CR11 CARD READER, WITH THE NECESSARY EXCEPTION THAT TIMING IN MOST CASES IS ONLY PARTIALLY TESTED. A SPECIAL TEST IS INCLUDED TO CHECK OUT THE ERROR FUNCTIONS OF THE G.D.I. 100 READER, WHICH PRINTS OUT DIRECTIONS AS IT GOES ALONG. A TEST IS ALSO INCLUDED TO ISOLATE DIFFICULT DATA ERRORS USING A SPECIAL TEST DECK PUNCHED BY THE USER.

10. LISTING

SEE FOLLOWING PAGES.

11. FLOWCHART

NOT APPLICABLE.

```
;DIAGNOSTIC FOR CR11 CARD READER
;COPYRIGHT 1970,1971,1972, DIGITAL EQUIPMENT CORP., MAYNARD, MASS, 01754
;BY RICK FADDEN
;(MODIFIED AUGUST-71 FOR DOCUMATION CARD READER (JOHN RODENHISER))
;(MODIFIED APRIL-72 FOR HARDWARE ECO)

;STARTING ADDRESSES ARE:
;       200=INSTRUCTION AND DATA TEST FOR THE CR11
;       210=ERROR FUNCTION TEST OF CR11 (GDI)
;       220=ERROR FUNCTION TEST OF CR11 USING DOCUMATION READER.
;       240=SINGLE TEST LOOP
;       250=READ SINGLE DATA PATTERN TEST

;SWITCH REGISTER SETTINGS FOR THE INSTRUCTION AND DATA TEST ARE:
;       SW04=1 FOR THE BINARY TEST DECK
;       SW05=1 TO HALT AT THE END OF A STANDARD 80 CARD
;              TEST DECK,
;              =0 TO CONTINUE FROM ONE DECK TO THE NEXT,
;                     AFTER THE LAST DECK IN THE HOPPER IS
;                     RUN, THE PROGRAM WAITS FOR THE CARD READER
;                     TO COME BACK ON-LINE,AND RUNS THRU
;                     A SERIES OF CHECKS OF OFF-LINE AND
;                     COMING ON-LINE OPERATIONS OF THE READER.
;                     WHEN THE READER IS BACK ON-LINE AND THE
;                     CHECKS ARE COMPLETE, THE DATA TEST IS RESUMED.
;       SW06=1 TO RUN THE COMBINED INSTRUCTION AND DATA TEST
;              WHEN CONTINUING FROM ONE DECK TO THE NEXT
;          =0 TO RUN ONLY THE DATA TEST ON EVERY DECK AFTER THE FIRST
;       SW07=1 TO RUN ONLY THE INSTRUCTION TEST CONTINUALLY
;              SETTING SW06 AND SW07 AT THE END OF A DECK WILL
;              CAUSE THE INSTRUCTION TEST TO BE RUN CONINUOUSLY FROM THEN ON
;              NOTE:  IF SW7 IS SET, CHECKED BY PROGRAM, AND THEN
;              CLEARED, THE DATA TEST WILL BE INCORRECT.  THIS IS
;              TRUE BECAUSE THE FIRST CARD IN THE DATA TEST WILL NOT
;              BE THE ONE EXPECTED.  WITH SW7 SET THE TEST MAY HANG
;              WHEN THE INPUT HOPPER RUNS OUT OF CARDS.
;       SW10=1 TO INDICATE THAT THE CR11 BEING TESTED USES THE
;              M829 MODULE
;          =0 TO INDICATE THAT THE CR11 BEING TESTED USES THE
;              M8290 MODULE
;       SW11=1 TO INHIBIT SUBPROGRAM ITERATION
;              (NOTE THAT IF PROGRAM FLOW IS ALLOWED TO ENTER THE
;              DATA SUBTEST, DATA ERRORS WILL OCCUR SINCE THE
;              CARD COUNT WILL BE INCORRECT.)
;       SW12=1 TO INHIBIT TRACE TRAPPING
;       SW13=1 TO INHIBIT PRINTOUT
;       SW14=1 FOR SCOPE LOOP
;       SW15=1 TO HALT ON ERROR

;OPERATING PROCEDURE FOR THE INSTRUCTION AND DATA TEST:
;       1.  LOAD TEST DECK IN CARD READER, PRESS "MOTOR START" AND "READ START"
;           ("RESET" ON THE DOCUMATION READER). IF THE DECK BEING USED
;           IS NOT THE STANDARD ALPHANUMERIC OR BINARY DECK, ONLY THE
;           INSTRUCTION PORTION OF THE TEST CAN BE RUN.
;           (SW7 MUST BE SET TO ONE TO INDICATE THIS).
```

```
;       2,  LOAD SA 200, THEN SET THE SWITCH REGISTER SWITCHES TO THE DESIRED
;           COMBINATION
;       3,  PRESS "START" ON THE CONSOLE
;       4,  NOTE THAT RUNNING THE COMPLETE INSTRUCTION TEST
;           REQUIRES THAT THE INPUT HOPPER MUST RUN OUT OF CARDS
;           AT THE END OF A TEST DECK AT LEAST ONCE.  WHEN THIS
;           OCCURS, THE PROCESSOR SHOULD CONTINUE TO RUN.  LOADING
;           A DECK INTO THE INPUT HOPPER AND PRESSING "MOTOR START" AND "READ
;           START" ("RESET" ON THE DOCUMATION READER) SHOULD CAUSE THE
;           BELL TO RING AND THE CARD READER TO RESUME READING CARDS.
;           IF THIS DOES NOT OCCUR, IT IS A FAULT AND SHOULD BE FIXED.

;SPECIAL SWITCH REGISTER SETTINGS FOR THE ERROR FUNCTION TEST:
;       SW00=1 TO INHIBIT TESTING THE DARK-LIGHT ERROR
;       SW14=1 TO LOOP THRU THE CURRENT SUBTEST
;       SW15=1 TO HALT ON ERROR

;OPERATING PROCEDURE FOR THE ERROR FUNCTION TEST:
;       1,  LOAD A FEW SPARE CARDS INTO THE INPUT HOPPER.
;       2,  PRESS "MOTOR START" AND "READ START" ON THE CARD READER
;           ("RESET" ON THE DOCUMATION READER).
;       3,  LOAD THE SA, THEN SET THE DESIRED SWITCH OPTIONS.
;       4,  PRESS "START" ON THE CONSOLE.
;       5,  FOLLOW THE INSTRUCTIONS AS THEY ARE PRINTED OUT.

;SINGLE TEST LOOP (SA 240) HALTS TWICE:
;       1ST HALT = LOAD STARTING ADDRESS OF DESIRED TEST (TEST1 TO TEST 24)
;       2ND HALT = SET SR OPTIONS (BIT 11 MUST = 0)
;       THIS TEST USES TRACE TRAPPING WHERE APPLICABLE IF SW12 IS NOT SET

;DESCRIPTION OF SINGLE DATA PATTERN TEST
;       THIS TEST IS DESIGNED TO AID IN THE LOCATION OF DIFFICULT DATA ERROR
;       PROBLEMS AND PERHAPS HELP IN SOME CARD READER ADJUSTMENTS.  IT
;       CONTINUOUSLY READS CARDS WHICH HAVE ALL COLUMNS PUNCHED IDENTICALLY,
;       CHECKING THE DATA AGINST A PATTERN SET UP ON THE SWITCHES INITIALLY.
;       ANY ERRORS ARE PRINTED OUT, ALONG WITH A COUNT OF THE TOTAL NUMBER
;       OF CARDS READ AND THE TOTAL NUMBER OF DATA ERRORS WHICH HAVE
;       OCCURRED SINCE THE TEST WAS STARTED.

;OPERATING PROCEDURE FOR SINGLE DATA PATTERN TEST:
;       1,  LOAD TEST DECK OF IDENTICAL CARDS IN THE INPUT HOPPER, AND PUT
;           THE CARD READER ON-LINE.
;       2,  LOAD SA 250, THEN PRESS "START" ON THE CONSOLE.
;       3,  AT THE INITIAL HALT SET THE CORRECT CARD-IMAGE
;           DATA PATTERN IN SW11-SW00, THEN PRESS CONTINUE.
;       4,  WHEN THE READER RUNS OUT OF CARDS IT WILL RING THE
;           BELL.  RELOADING THE DECK AND PRESSING "MOTOR START" AND
;           "READ START" ("RESET" ON THE DOCUMATION READER) WILL
;           CONTINUE THE TEST.

177570          SR=177570
177776          PSR=177776
000240          NOP=240
104000          HLT=EMT
104400          SCOPE=TRAP
```

```
                    000000                  ADINT=%0                ;CONTAINS ADDRESS OF INTERRUPT VECTOR
                    000001                  COUNT=%1                ;USED FOR TIMING, ETC.
                    000002                  R2=%2                   ;SCRATCH
                    000003                  CRS=%3                  ;CONTAINS ADDRESS OF CARD READ STATUS REGISTER
                    000004                  CRB1=%4                 ;CONTAINS ADDRESS OF CARD READER BUFFER (12 BIT DATA)
                    000005                  R5=%5                   ;SCRATCH
                    000006                  SP=%6                   ;STACK POINTER
                    000007                  PC=%7                   ;PROGRAM COUNTER
                                        ;LOAD TRAP CATCHER INTO LOCATIONS 0 THRU 377
                    000100                  .REPT 100
                                            .+2
                                            HALT
                                            .ENDR
000000  000002                            .+2
000002  000000                            HALT
000004  000006                            .+2
000006  000000                            HALT
000010  000012                            .+2
000012  000000                            HALT
000014  000016                            .+2
000016  000000                            HALT
000020  000022                            .+2
000022  000000                            HALT
000024  000026                            .+2
000026  000000                            HALT
000030  000032                            .+2
000032  000000                            HALT
000034  000036                            .+2
000036  000000                            HALT
000040  000042                            .+2
000042  000000                            HALT
000044  000046                            .+2
000046  000000                            HALT
000050  000052                            .+2
000052  000000                            HALT
000054  000056                            .+2
000056  000000                            HALT
000060  000062                            .+2
000062  000000                            HALT
000064  000066                            .+2
000066  000000                            HALT
000070  000072                            .+2
000072  000000                            HALT
000074  000076                            .+2
000076  000000                            HALT
000100  000102                            .+2
000102  000000                            HALT
000104  000106                            .+2
000106  000000                            HALT
000110  000112                            .+2
000112  000000                            HALT
000114  000116                            .+2
000116  000000                            HALT
000120  000122                            .+2
000122  000000                            HALT
```

```
000124  000126                          .+2
000126  000000                          HALT
000130  000132                          .+2
000132  000000                          HALT
000134  000136                          .+2
000136  000000                          HALT
000140  000142                          .+2
000142  000000                          HALT
000144  000146                          .+2
000146  000000                          HALT
000150  000152                          .+2
000152  000000                          HALT
000154  000156                          .+2
000156  000000                          HALT
000160  000162                          .+2
000162  000000                          HALT
000164  000166                          .+2
000166  000000                          HALT
000170  000172                          .+2
000172  000000                          HALT
000174  000176                          .+2
000176  000000                          HALT
000200  000202                          .+2
000202  000000                          HALT
000204  000206                          .+2
000206  000000                          HALT
000210  000212                          .+2
000212  000000                          HALT
000214  000216                          .+2
000216  000000                          HALT
000220  000222                          .+2
000222  000000                          HALT
000224  000226                          .+2
000226  000000                          HALT
000230  000232                          .+2
000232  000000                          HALT
000234  000236                          .+2
000236  000000                          HALT
000240  000242                          .+2
000242  000000                          HALT
000244  000246                          .+2
000246  000000                          HALT
000250  000252                          .+2
000252  000000                          HALT
000254  000256                          .+2
000256  000000                          HALT
000260  000262                          .+2
000262  000000                          HALT
000264  000266                          .+2
000266  000000                          HALT
000270  000272                          .+2
000272  000000                          HALT
000274  000276                          .+2
000276  000000                          HALT
000300  000302                          .+2
```

```
000302  000000                  HALT
000304  000306                  .+2
000306  000000                  HALT
000310  000312                  .+2
000312  000000                  HALT
000314  000316                  .+2
000316  000000                  HALT
000320  000322                  .+2
000322  000000                  HALT
000324  000326                  .+2
000326  000000                  HALT
000330  000332                  .+2
000332  000000                  HALT
000334  000336                  .+2
000336  000000                  HALT
000340  000342                  .+2
000342  000000                  HALT
000344  000346                  .+2
000346  000000                  HALT
000350  000352                  .+2
000352  000000                  HALT
000354  000356                  .+2
000356  000000                  HALT
000360  000362                  .+2
000362  000000                  HALT
000364  000366                  .+2
000366  000000                  HALT
000370  000372                  .+2
000372  000000                  HALT
000374  000376                  .+2
000376  000000                  HALT
                        ;LOAD TRAP VECTORS FOR HLT AND SCOPE ROUTINES
        000014                  .=14
000014  000622                  THTRAP
000016  000340                  340
        000030                  .=30
000030  011434                  PRINT
000032  000340                  340
000034  011716                  SCOPEC
000036  000340                  340
                        ;LOAD STARTING ADDRESS AREA
        000200                  .=200
000200  012706  000600          MOV     #STACK,SP
000204  000167  000464          JMP     BEGIN           ;NORMAL STARTING ADDRESS FOR G,D,I, 100 READER
000210  012706  000600          MOV     #STACK,SP
000214  000167  000664          JMP     ERCR11          ;STARTING ADDRESS FOR CR11 ERROR FUNCTION TEST (G,D,I)
000220  012706  000600          MOV     #STACK,SP
000224  000167  000662          JMP     ERCM11          ;STARTING ADDRESS FOR CR11 ERROR FUNCTION TEST (DOCUMATION)
        000240                  .=240
000240  012706  000600          MOV     #STACK,SP
000244  000167  010324          JMP     TESTX           ;STARTING ADDRESS FOR LOOP WHICH CONTINUALLY RUNS
                                                        ;ANY SINGLE SUBTEST
000250  012706  000600          MOV     #STACK,SP
000254  000167  010416          JMP     CKSAME          ;STARTING ADDRESS OF TEST TO READ A SINGLE DATA
                                                        ;PATTERN CONTINUOUSLY
```

```
                                ;LOAD POINTERS AND GENERAL STORAGE
        000600                          .=600
000600  000000          STACK:  0               ;STACK POINTER INITIALIZED TO POINT HERE
000602  000000          INTFLG: 0               ;CONTAINS LEVEL THAT INTERRUPT IS FOUND AT
000604  000230          INTVC:  230             ;ADDRESS OF CARD READER INTERRUPT VECTOR
000606  177564          TCSR:   177564          ;ADDRESS OF TELETYPE STATUS REGISTER
000610  177566          TDBR:   177566          ;ADDRESS OF TELETYPE DATA BUFFER
000612  000000          FLAG:   0               ;SET TO ONE FOR MARK-SENSE CARD READER
000614  177160          KCRS:   177160          ;ADDRESS OF CARD READER STATUS REGISTER
000616  177162          KCRB1:  177162          ;ADDRESS OF CARD READER DATA BUFFER
000620  177164          CRB2:   177164          ;ADDRESS TO READ ENCODED DATA
000622  000002          TRTRAP: RTI             ;RETURN FROM TRACE LOOP
000624  000000          TRFLG:  0               ;TOGGLED TO SWITCH BETWEEN TRACE TRAPPING AND NORMAL FLOW
000626  000000          PROC:   0               ;STORES PROCESSOR STATUS WHEN TRACE TRAP MUST BE CLEARED
                                                ;IN A SUBTEST
000630  000000          ERFLG:  0               ;SET TO ZERO TO OUTPUT DATA ERROR HEADING


                                ;INITIALIZE CSR AND DBR POINTERS
000632  012767  000001  011136  SETUP:  MOV     #1,ITMAX        ;SET ITERATION MAXIMUM TO 1 ITERATION
000640  016703  177750          MOV     KCRS,CRS        ;SET UP REGISTER POINTERS
000644  016704  177746          MOV     KCRB1,CRB1
000650  016700  177730          MOV     INTVC,ADINT     ;LOAD ADDRESS OF INTERRUPT VECTOR
000654  005067  177722          CLR     INTFLG          ;INITIALIZE INTERRUPT FLAG
000660  005067  177740          CLR     TRFLG           ;INITIALIZE TRACE FLAG
000664  012767  000340  177104  BEGIN:  MOV     #340,PSR        ;SETUP PROCESSOR STATUS
000672  000207          RTS     %7              ;RETURN
000674  004767  177732          JSR     %7,SETUP        ;INITIALIZE POINTERS AND FLAGS
000700  000416          BR      TEST            ;GO TO INSTRUCTION TESTS
000702  005767  177716  RESTRT: TST     TRFLG           ;CHECK FOR TRACE TRAPPING
000706  001004          BNE     TRAPX           ;IF SET, TRACE TRAP
000710  012767  000340  177060  NOTRP:  MOV     #340,PSR        ;IF ZERO, CLEAR TRACE BIT
000716  000407          BR      TEST            ;GO TO INSTRUCTION TESTS
000720  032767  010000  176642  TRAPX:  BIT     #10000,SR       ;CHECK SW12
000726  001370          BNE     NOTRP           ;BRANCH IF SET TO CLEAR TRACE BIT
000730  012767  000360  177040          MOV     #360,PSR        ;SET TRACE BIT

                                ;TEST FOR CORRECT INITIALIZATION OF STATUS REGISTER
000736  012767  000746  011036  TEST:   MOV     #TEST1A,RETURN  ;SETUP SCOPE LOOP RETURN ADDRESS
000744  104400          TEST1:  SCOPE
000746  004767  010374  TEST1A: JSR     %7,CKBIT8       ;CHECK FOR OFF-LINE SET
000752  016767  177020  177646          MOV     PSR,PROC        ;STORE PROCESSOR STATUS
000760  005067  177012          CLR     PSR             ;CLEAR TRACE BIT
000764  005001          CLR     COUNT           ;INITIALIZE COUNTER
000766  005201          INC     COUNT           ;WAIT TO BE CERTAIN
000770  001376          BNE     .-2             ;THAT ALL CARDS ARE
000772  005201          INC     COUNT           ;THRU BEFORE ISSUING
000774  001376          BNE     .-2             ;INIT
000776  016767  177624  176772          MOV     PROC,PSR        ;RESTORE PROCESSOR STATUS
001004  000005          RESET                   ;SEND OUT INIT
001006  005713          TST     @CRS            ;CHECK FOR STATUS REGISTER ALL ZERO
001010  001401          BEQ     .+4             ;BRANCH IF OK
001012  104000          HLT                     ;STATUS REGISTER NOT CORRECTLY INITIALIZED
                                ;ONLY BITS 1 AND 6 OF THE STATUS REGISTER SHOULD BE ABLE TO BE SET TO ONE
                                ;AND READ BACK AS ONE
```

```
001014  052713  177776                  BIS     #177776,@CRS    ;SET ALL BITS BUT 0
001020  022713  000102                  CMP     #102,@CRS       ;ONLY BITS 1 AND 6 SHOULD BE SET
001024  001402                          BEQ     .+6             ;BRANCH IF OK
001026  104000                          HLT                     ;STATUS REGISTER DIDN'T CONTAIN 102
001030  000404                          BR      TEST2           ;BRANCH AFTER FAILURE
                                        ;CLEARING STATUS REGISTER SHOULD CLEAR BITS 1 AND 6
001032  005013                          CLR     @CRS            ;CLEAR BITS 1 AND 6
001034  005713                          TST     @CRS            ;CHECK FOR ALL BITS CLEAR
001036  001401                          BEQ     .+4             ;BRANCH IF OK
001040  104000                          HLT                     ;BIT 1 AND/OR BIT 6 DIDN'T CLEAR

001042  104400                  TEST2:  SCOPE
                                        ;START SHOULD CAUSE CARD DONE WITHIN 1 SECOND
                                        ;BIT 0 SHOULD ALWAYS READ AS BEING EQUAL TO ZERO
001044  004767  010276                  JSR     %7,CKBIT8       ;CHECK FOR OFF-LINE SET
001050  016767  176722  177550          MOV     PSR,PROC        ;STORE CURRENT PROCESSOR STATUS
001056  005067  176714                  CLR     PSR             ;CLEAR TRACE BIT
001062  005213                          INC     @CRS            ;START READING A CARD
001064  032713  000001                  BIT     #1,@CRS         ;CHECK BIT 0
001070  001401                          BEQ     .+4             ;BRANCH IF NOT SET
001072  104000                          HLT                     ;BIT 0 READ AS A ONE
001074  005227  000000                  INC     #0              ;WAIT
001100  001375                          BNE     .-4
001102  005227  000000                  INC     #0
001106  001375                          BNE     .-4
001110  005227  000000                  INC     #0
001114  001375                          BNE     .-4
001116  005227  000000                  INC     #0
001122  001375                          BNE     .-4
001124  005227  000000                  INC     #0
001130  001375                          BNE     .-4
001132  016767  177470  176636          MOV     PROC,PSR        ;RESTORE PROCESSOR STATUS
001140  032713  040000                  BIT     #40000,@CRS     ;CHECK CARD DONE
001144  001002                          BNE     CONT2           ;CONTINUE IF SET
001146  104000                          HLT                     ;CARD DONE DIDN'T SET WITHIN 400 MS
001150  000406                          BR      TEST3           ;NOTE THAT FAILURE COULD BE DUE TO READ
                                                                ;NOT BEING RESET
001152  052713  040000          CONT2:  BIS     #40000,@CRS     ;DATO TO STATUS REGISTER SHOULD CLEAR
001156  032713  040000                  BIT     #40000,@CRS     ;CARD DONE
001162  001401                          BEQ     .+4             ;BRANCH IF IT DID
001164  104000                          HLT                     ;DATO DIDN'T CLEAR CARD DONE

001166  104400                  TEST3:  SCOPE
                                        ;BUSY (BIT 9) SHOULD BE SET BY READING A CARD
                                        ;IT SHOULD REMAIN SET UNTIL CARD DONE SETS, WHICH SHOULD CLEAR IT
001170  004767  010152                  JSR     %7,CKBIT8       ;CHECK FOR OFF-LINE SET
001174  005013                          CLR     @CRS            ;INITIALIZE STATUS REGISTER
001176  005213                          INC     @CRS            ;READ A CARD
001200  032713  001000                  BIT     #1000,@CRS      ;CHECK BUSY
001204  001002                          BNE     LOOP3           ;BRANCH IF SET
001206  104000                          HLT                     ;READING A CARD DIDN'T SET BUSY
001210  000417                          BR      TEST4
001212  032713  040000          LOOP3:  BIT     #40000,@CRS     ;CHECK CARD DONE
001216  001010                          BNE     DONE3           ;BRANCH IF SET
001220  032713  001000                  BIT     #1000,@CRS      ;CHECK BUSY
```

```
001224  001372                  BNE     LOOP3           ;BRANCH IF STILL SET
001226  032713  040000          BIT     #40000,@CRS     ;CHECK CARD DONE
001232  001006                  BNE     TEST4           ;GO TO NEXT TEST IF SET
001234  104000                  HLT                     ;BUSY CLEARED BEFORE CARD DONE SET
001236  000404                  BR      TEST4
001240  032713  001000  DONE3:  BIT     #1000,@CRS      ;CHECK BUSY
001244  001401                  BEQ     TEST4           ;GO ON TO NEXT TEST IF CLEAR
001246  104000                  HLT                     ;CARD DONE DIDN'T CLEAR BUSY

001250  104400          TEST4:  SCOPE
                        ;A TIMING ERROR SHOULD OCCUR IF DATA IS NOT READ AND NEW DATA COMES IN
                        ;A TIMING ERROR SHOULD SET THE SPECIAL CONDITION BIT WHEN CARD DONE OCCURS
                        ;COLUMN READY SHOULD BE CLEARED BY THE TIMING ERROR AND PREVENTED FROM RESETTING
                        ;BITS 11,14, AND 15 SHOULD BE CLEARED BY A DATO TO THE STATUS REGISTER
001252  004767  010016          JSR     %7,INIT         ;INITIALIZE STATUS REGISTER

001256  005001                  CLR     COUNT           ;INITIALIZE COUNTER
001260  005213                  INC     @CRS            ;INITIATE READ
001262  032713  140200  LOOP4:  BIT     #140200, @CRS   ;WAIT FOR SPECIAL CONDITION, CARD DONE,
                                                        ;OR COLUMN READY
001266  001775                  BEQ     LOOP4           ;LOOP IF NONE OCCURRED
001270  032713  140000          BIT     #140000, @CRS   ;SPECIAL CONDITION OR CARD DONE?
001274  001007                  BNE     CK4             ;YES, BRANCH
001276  005201                  INC     COUNT           ;NO, COUNT COLUMN READYS
001300  105713  LOOP4B: TSTB    @CRS                    ;WAIT FOR COLUMN READY TO CLEAR
001302  100367                  BPL     LOOP4           ;IF CLEAR, RETURN TO LOOP4
001304  032713  140000          BIT     #140000, @CRS   ;CHECK FOR SPECIAL CONDITION OR CARD DONE
001310  001001                  BNE     CK4             ;BRANCH IF EITHER SET
001312  000772                  BR      LOOP4B          ;OTHERWISE, CHECK AGAIN
001314  032713  040000  CK4:    BIT     #40000, @CRS    ;CHECK CARD DONE
001320  001002                  BNE     .+6             ;BRANCH IF SET
001322  104000                  HLT                     ;SPECIAL CONDITION SET BEFORE CARD DONE
001324  000403                  BR      CONT4
001326  005713                  TST     @CRS            ;CHECK SPECIAL CONDITION
001330  100401                  BMI     .+4             ;BRANCH IF SET
001332  104000                  HLT                     ;SPECIAL CONDITION WASN'T  SET
001334  032713  004000  CONT4:  BIT     #4000,  @CRS    ;CHECK TIMING ERROR
001340  001001                  BNE     .+4             ;BRANCH IF SET
001342  104000                  HLT                     ;TIMING ERROR WASN'T SET
001344  005301                  DEC     COUNT           ;CHECK NUMBER OF COLUMN READYS
001346  100002                  BPL     .+6             ;BRANCH IF ANY OCCURRED
001350  104000                  HLT                     ;COLUMN READY NEVER OCCURRED
001352  000402                  BR      .+6
001354  001401                  BEQ     .+4             ;BRANCH IF ONLY ONE OCCURRED
001356  104000                  HLT                     ;COLUMN READY OCCURRED MORE THAN ONCE
001360  105713                  TSTB    @CRS            ;CHECK COLUMN READY
001362  100001                  BPL     .+4             ;BRANCH IF NOT SET
001364  104000                  HLT                     ;COLUMN READY WASN'T CLEARED
001366  005013                  CLR     @CRS            ;CLEAR BITS 11,14, AND 15 VIA DATO
001370  032713  144000          BIT     #144000, @CRS   ;CHECK
001374  001401                  BEQ     .+4
001376  104000                  HLT                     ;BITS 11,14, AND 15 WEREN'T ALL CLEARED


001400  104400          TEST5:  SCOPE
```

```
                                ;SETTING READ SHOULD CAUSE COLUMN READY TO SET 80 TIMES BEFORE CARD DONE SETS
                                ;READING THE DATA BUFFER SHOULD CLEAR COLUMN READY AND PREVENT A TIMING ERROR
001402  004767  007666                  JSR     %7,INIT         ;INITIALIZE STATUS REGISTER
001406  005001                          CLR     COUNT           ;INITIALIZE COUNTER
001410  005213                          INC     @CRS            ;INITIATE READ
001412  032713  140200          LOOP5:  BIT     #140200,@CRS    ;WAIT FOR COLUMN READY, CARD DONE
001416  001775                          BEQ     .-4             ;OR SPECIAL CONDITION
001420  032713  040000                  BIT     #40000, @CRS    ;CARD DONE?
001424  001015                          BNE     CK5             ;YES, BRANCH
001426  005713                          TST     @CRS            ;CHECK BIT 15
001430  100002                          BPL     .+6             ;SKIP ERROR HALT IF NOT SET
001432  104000                          HLT                     ;BIT 15 WAS SET
001434  000437                          BR      TEST6           ;GO TO NEXT TEST
001436  020127  000117                  CMP     COUNT,  #79.    ;CHECK FOR 80
001442  100363                          BPL     LOOP5           ;BRANCH IF 80 OR MORE WITHOUT CLEARING READY
001444  005201                          INC     COUNT           ;INCREMENT COUNTER
001446  005714                          TST     @CRB1           ;CLEAR READY
001450  105713                          TSTB    @CRS            ;MAKE SURE IT CLEARED
001452  100001                          BPL     .+4             ;BRANCH IF IT DID
001454  104000                          HLT                     ;READING DATA BUFFER DIDN'T CLEAR READY
001456  000755                          BR      LOOP5           ;LOOP
001460  032713  004000          CK5:    BIT     #4000,  @CRS    ;CHECK TIMING ERROR BIT
001464  001401                          BEQ     .+4             ;BRANCH IF NOT SET
001466  104000                          HLT                     ;TIMING ERROR WAS SET
                                ;NOTE THAT IF COLUMN READY SET MORE THAN 80 TIMES
                                ;A TIMING ERROR WILL OCCUR AND THE COUNT WILL BE 79 (=117 OCTAL)
001470  000421                          BR      TEST6           ;BRANCH AFTER ERROR
001472  020127  000117                  CMP     COUNT,  #79.    ;CHECK COUNT
001476  001401                          BEQ     .+4             ;BRANCH IF 80 COLUMN READYS OCCURRED
001500  104000                          HLT                     ;COLUMN READY DIDN'T OCCUR 80 TIMES
                                                                ;BEFORE CARD DONE
001502  021327  040200                  CMP     @CRS,#40200     ;ONLY CARD DONE AND COLUMN READY SHOULD BE SET
001506  001401                          BEQ     .+4
001510  104000                          HLT                     ;STATUS REGISTER IN WRONG STATE
001512  005013                          CLR     @CRS            ;SHOULD CLEAR DONE BUT NOT READY
001514  021327  000200                  CMP     @CRS,#200       ;CHECK FOR ONLY READY SET
001520  001401                          BEQ     .+4             ;BRANCH IF OK
001522  104000                          HLT                     ;STATUS REGISTER IN WRONG STATE
001524  005714                          TST     @CRB1           ;READING DATA BUFFER SHOULD CLEAR COLUMN READY
001526  005713                          TST     @CRS            ;CHECK STATUS REGISTER
001530  001401                          BEQ     .+4             ;BRANCH IF ALL BITS ZERO
001532  104000                          HLT                     ;STATUS REGISTER NOT EQUAL TO ZERO

001534  104400          TEST6:  SCOPE
                                ;A TIMING ERROR SHOULD SET BIT 11 BEFORE CARD DONE OCCURS, EVEN IF IT OCCURS AT COLUMN 80
                                ;A DATOB TO THE LOW BYTE OF THE CRS SHOULD CLEAR BITS 15,14, AND 11
001536  004767  007532                  JSR     %7,INIT         ;INITIALIZE
001542  012701  000115                  MOV     #77.,,COUNT     ;SETUP COUNTER
001546  005213                          INC     @CRS            ;START READING A CARD
001550  105713          LOOP6:  TSTB    @CRS            ;WAIT FOR COLUMN READY
001552  100376                          BPL     .-2
001554  005714                          TST     @CRB1           ;CLEAR COLUMN READY
001556  005301                          DEC     COUNT           ;GO THRU LOOP FOR 1ST 78 COLUMN READY'S
001560  100373                          BPL     LOOP6
001562  032713  144000                  BIT     #144000,@CRS    ;WAIT FOR CARD DONE OR TIMING ERROR
```

```
001566  001775                  BEQ     .-4             ;OR SPECIAL CONDITION
001570  032713  040000          BIT     #40000,@CRS     ;CARD DONE SET?
001574  001026                  BNE     ERR6            ;YES, 2 POSSIBLE TEST FAILURES
001576  032713  004000          BIT     #4000,@CRS      ;CHECK TIMING ERROR
001602  001416                  BEQ     OFF6            ;IF NOT SET, READER IS PROBABLY OFF-LINE
001604  105713                  TSTB    @CRS            ;CHECK COLUMN READY
001606  100001                  BPL     .+4             ;BRANCH IF CLEAR
001610  104000                  HLT                     ;TIMING ERROR DIDN'T CLEAR READY
001612  005713                  TST     @CRS            ;WAIT FOR SPECIAL CONDITION
001614  100376                  BPL     .-2
001616  032713  040000          BIT     #40000,@CRS     ;CHECK CARD DONE
001622  001406                  BEQ     OFF6            ;IF NOT SET, READER IS PROBABLY OFF-LINE
001624  105013                  CLRB    @CRS            ;DATOB TO LOW BYTE OF CRS
001626  032713  144000          BIT     #144000,@CRS    ;CHECK BITS 15,14,11
001632  001415                  BEQ     TEST7           ;BRANCH IF CLEAR TO NEXT TEST
001634  104000                  HLT                     ;DATOB TO LOW BYTE OF CRS DIDN'T CLEAR
                                                        ;BITS 15,14 AND/OR 11
001636  000413                  BR      TEST7           ;GO TO NEXT TEST
001640  032713  000400  OFF6:   BIT     #400,@CRS       ;CHECK BIT 8
001644  001010                  BNE     TEST7           ;BRANCH IF SET
001646  104000                  HLT                     ;BIT 15 WAS SET, 8 WASN'T
001650  000406                  BR      TEST7           ;GO TO NEXT TEST
001652  032713  004000  ERR6:   BIT     #4000,@CRS      ;TIMING ERROR SET?
001656  001402                  BEQ     .+6             ;NO, BRANCH
001660  104000                  HLT                     ;TIMING ERROR DIDN'T SET BEFORE CARD DONE
001662  000401                  BR      TEST7           ;GO TO NEXT TEST AFTER ERROR
001664  104000                  HLT                     ;TIMING ERROR WASN'T SET

001666  104400          TEST7:  SCOPE
                        ;NOT READING THE EIGHTIETH COLUMN OF DATA FROM THE BUFFER
                        ;SHOULD CAUSE A TIMING ERROR ON THE FIRST COLUMN OF THE NEXT CARD
                        ;SETTING EJECT SHOULD CLEAR TIMING ERROR, AND BIT 15 SHOULDN'T SET
                        ;INCB SHOULD START A READ

001670  004767  007400          JSR     %7,INIT         ;INITIALIZE
001674  005213                  INC     @CRS            ;START READ
001676  012701  000120          MOV     #80.,,COUNT     ;INITIALIZE COUNTER
001702  032713  140200  LOOP7:  BIT     #140200,@CRS    ;TEST FOR ERROR, DONE OR READY
001706  001775                  BEQ     LOOP7           ;LOOP IF NONE SET
001710  005713                  TST     @CRS            ;CHECK ERROR
001712  100002                  BPL     .+6             ;BRANCH IF NOT SET
001714  104000                  HLT                     ;BIT 15 WAS SET
001716  000455                  BR      TEST8           ;GO TO NEXT TEST AFTER ERROR
001720  032713  040000          BIT     #40000,@CRS     ;CHECK FOR CARD DONE
001724  001013                  BNE     DONE7           ;BRANCH IF SET
001726  005301                  DEC     COUNT           ;COUNT
001730  001402                  BEQ     .+6             ;IF 80TH COLUMN READY, BRANCH
001732  005714                  TST     @CRB1           ;CLEAR DONE
001734  000762                  BR      LOOP7           ;LOOP
001736  032713  140000          BIT     #140000,@CRS    ;WAIT FOR DONE OR SPECIAL CONDITION
001742  001775                  BEQ     .-4
001744  005713                  TST     @CRS            ;CHECK SPECIAL CONDITION
001746  100002                  BPL     DONE7           ;BRANCH IF NOT SET
001750  104000                  HLT                     ;SPECIAL CONDITION WAS SET
001752  000437                  BR      TEST8           ;GO TO NEXT TEST AFTER ERROR
```

```
001754  005701                  DONE7:  TST     COUNT           ;TEST FOR 80 COLUMN READY'S
001756  001402                          BEQ     .+6             ;BRANCH IF OK
001760  104000                          HLT                     ;COLUMN READY DID NOT OCCUR 80 TIMES
001762  000433                          BR      TEST8           ;GO TO NEXT TEST AFTER ERROR
001764  105213                          INCB    @CRS            ;START READ
001766  105713                          TSTB    @CRS            ;CHECK COLUMN READY
001770  100401                          BMI     .+4             ;BRANCH IF STILL SET
001772  104000                          HLT                     ;READY DID NOT REMAIN SET
001774  032713  004000                  BIT     #4000,@CRS      ;TEST FOR TIMING ERROR
002000  001775                          BEQ     .-4             ;LOOP IF NOT SET
002002  105713                          TSTB    @CRS            ;CHECK COLUMN READY
002004  100002                          BPL     .+6             ;BRANCH IF NOT SET
002006  104000                          HLT                     ;TIMING ERROR DIDN'T CLEAR READY
002010  000420                          BR      TEST8
002012  112713  000002                  MOVB    #2,@CRS         ;SET EJECT
002016  032713  004000                  BIT     #4000,@CRS      ;CHECK TIMING ERROR
002022  001402                          BEQ     .+6             ;BRANCH IF CLEARED
002024  104000                          HLT                     ;TIMING ERROR NOT CLEARED BY DATOB
002026  000411                          BR      TEST8           ;GO TO NEXT TEST AFTER ERROR
002030  032713  140000                  BIT     #140000,@CRS    ;WAIT FOR DONE OR SPECIAL CONDITION
002034  001775                          BEQ     .-4
002036  032713  000400                  BIT     #400,@CRS       ;CHECK BIT 8
002042  001003                          BNE     TEST8           ;BRANCH IF READER OFF-LINE
002044  005713                          TST     @CRS            ;SPECIAL CONDITION SHOULDN'T SET
002046  100001                          BPL     .+4             ;SINCE DATOB CLEARED TIMING ERROR
002050  104000                          HLT                     ;BIT 15 WAS SET, 8 WASN'T

002052  104400                  TEST81: SCOPE
                                ;DATA SHOULD BE AVAILABLE IN THE DATA BUFFER FOR AT LEAST 1.0 MILLISECOND
002054  004767  007214                  JSR     %7,INIT         ;INITIALIZE STATUS REGISTER
002060  016767  175712  176540          MOV     PSR,PROC        ;STORE CURRENT PROCESSOR STATUS
002066  005067  175704                  CLR     PSR             ;CLEAR TRACE BIT
002072  005213                          INC     @CRS            ;START READ
002074  032713  140200          LOOP8:  BIT     #140200,@CRS    ;WAIT FOR COLUMN READY OR CARD DONE
002100  001775                          BEQ     .-4             ;OR SPECIAL CONDITION
002102  032713  040000                  BIT     #40000,@CRS     ;CARD DONE?
002106  001023                          BNE     DBRCK8          ;YES, GO TO CHECK STROBING OF DBR
002110  005713                          TST     @CRS            ;NO, CHECK BIT 15
002112  100002                          BPL     .+6             ;BRANCH IF NOT SET
002114  104000                          HLT                     ;BIT 15 WAS SET
002116  000441                          BR      TEST9           ;GO TO NEXT TEST AFTER ERROR
002120  005013                          CLR     @CRS            ;DATO TO CRS - SHOULDN'T CLEAR BUSY OR READY
002122  022713  001200                  CMP     #1200,@CRS      ;CHECK FOR BUSY AND READY
002126  001402                          BEQ     .+6             ;BRANCH IF STILL SET
002130  104000                          HLT                     ;CRS IN WRONG STATE
002132  000433                          BR      TEST9           ;GO TO NEXT TEST AFTER ERROR
002134  011405                          MOV     @CRB1,R5        ;STORE DATA
002136  012701  000300                  MOV     #300,COUNT      ;INITIALIZE COUNTER
002142  005301                          DEC     COUNT           ;WAIT FOR 1 MILLISECOND (APPROX.)
002144  001376                          BNE     .-2
002146  021405                          CMP     @CRB1,R5        ;DATA UNCHANGED?
002150  001751                          BEQ     LOOP8           ;OK, CONTINUE
002152  104000                          HLT                     ;DATA NOT AVAILABLE FOR 1.0 MILLISECONDS
002154  000422                          BR      TEST9           ;GO TO NEXT TEST AFTER FAILURE
```

```
002156  017702  176436          DBRCK8: MOV     @CRB2,R2        ;STORE ENCODED DATA IN REGISTER 2
002162  012701  000100                  MOV     #100,COUNT      ;SET UP COUNTER
002166  021405                  CONT8:  CMP     @CRB1,R5        ;READ CARD-IMAGE DATA BUFFER
002170  001402                          BEQ     .+6             ;BRANCH IF UNCHANGED
002172  104000                          HLT                     ;CRB1 READ INCORRECTLY
002174  000437                          BR      REST8           ;BRANCH TO RESTORE PROCESSOR STATUS AND EXIT
002176  027702  176416                  CMP     @CRB2,R2        ;READ ENCODED DATA BUFFER
002202  001402                          BEQ     .+6             ;BRANCH IF UNCHANGED
002204  104000                          HLT                     ;CRB2 READ INCORRECTLY
002206  000432                          BR      REST8           ;BRANCH AFTER FAILURE
002210  005301                          DEC     COUNT           ;COUNT DOWN
002212  001365                          BNE     CONT8           ;LOOP IF NOT DONE
002214  016767  176406  175554  REST8:  MOV     PROC,PSR        ;RESTORE PROCESSOR STATUS


002222  104400                  TEST9:  SCOPE
                                ;EJECT SHOULD PREVENT FURTHER COLUMN READY'S
                                ;CARD DONE SHOULD STILL OCCUR, AND TIMING ERRORS SHOULD BE
                                ;PREVENTED IF THE CURRENT COLUMN READY IS CLEARED
002224  004767  007044                  JSR     %7,INIT         ;INITIALIZE STATUS REGISTER
002230  016767  175542  176370          MOV     PSR,PROC        ;SAVE PROCESSOR STATUS
002236  005067  175534                  CLR     PSR             ;CLEAR TRACE BIT
002242  005213                          INC     @CRS            ;START READ
002244  105713                          TSTB    @CRS            ;WAIT FOR COLUMN READY
002246  001776                          BEQ     .-2
002250  052713  000002                  BIS     #2,@CRS         ;SET EJECT
002254  005714                          TST     @CRB1           ;CLEAR COLUMN READY
002256  005001                          CLR     COUNT           ;LOOP TAKES 11,4 MICROSECONDS ONCE THRU
002260  032713  044200          WAIT9:  BIT     #44200,@CRS     ;WAIT FOR CARD DONE, TIMING ERROR, OR
002264  001004                          BNE     CK9             ;COLUMN READY
002266  005201                          INC     COUNT           ;TIME FOR ABOUT 3/4 SECOND
002270  001373                          BNE     WAIT9           ;CONTINUE WAITING
002272  104000                          HLT                     ;NO CARD DONE OCCURRED WITHIN 3/4 SECOND
002274  000411                          BR      REST9           ;CONTINUE AFTER FAILURE
002276  032713  040000          CK9:    BIT     #40000,@CRS     ;CHECK FOR CARD DONE
002302  001006                          BNE     REST9
002304  032713  000200                  BIT     #200,@CRS       ;CHECK COLUMN READY
002310  001402                          BEQ     .+6             ;BRANCH IF NOT SET
002312  104000                          HLT                     ;COLUMN READY WAS SET
002314  000421                          BR      REST9
002316  104000                          HLT                     ;EJECT DID NOT PREVENT A TIMING ERROR
002320  016767  176302  175450  REST9:  MOV     PROC,PSR        ;RESTORE PROCESSOR STATUS


002326  104400                  TEST10: SCOPE
                                ;CARD DONE SHOULD CAUSE AN INTERRUPT
002330  004767  006740                  JSR     %7,INIT         ;INITIALIZE
002334  012710  002410                  MOV     #TINT10,@ADINT  ;LOAD RETURN POINTER
002340  052767  000340  175430          BIS     #340,PSR        ;SET PROCESSOR TO LEVEL 7
002346  016760  175424  000002          MOV     PSR,2(ADINT)    ;LOAD RETURN PROCESSOR STATUS
002354  042767  000340  175414          BIC     #340,PSR        ;SET PROCESSOR PRIORITY TO 0
002362  012713  000103                  MOV     #103,@CRS       ;SET EJECT, INTERRUPT ENABLE, AND READ
002366  032713  040000                  BIT     #40000,@CRS     ;WAIT FOR CARD DONE
002372  001775                          BEQ     .-4
002374  016067  000002  175374          MOV     2(ADINT),PSR    ;RESTORE PROCESSOR TO HIGHEST PRIORITY
```

```
002402  105013                          CLRB    @CRS            ;CLEAR INTERRUPT ENABLE
002404  104000                          HLT                     ;NO INTERRUPT OCCURRED
002406  000414                          BR      CONT10
002410  032713  040000          TINT10: BIT     #40000,@CRS     ;CHECK CARD DONE
002414  001001                          BNE     .+4             ;BRANCH IF SET
002416  104000                          HLT                     ;CARD DONE NOT SET
002420  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
002422  005713                          TST     @CRS            ;MAKE SURE NO ERROR OCCURRED
002424  100001                          BPL     .+4
002426  104000                          HLT                     ;BIT 15 WAS SET
002430  105713                          TSTB    @CRS            ;CHECK COLUMN READY
002432  100001                          BPL     .+4             ;BRANCH IF NOT SET
002434  104000                          HLT                     ;COLUMN READY WAS SET
002436  005013                          CLR     @CRS            ;DISABLE INTERRUPTS
002440  012710  000232          CONT10: MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
002444  005037  000232                  CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS

002450  104470                  TEST11: SCOPE
                                ;COLUMN READY SHOULD CAUSE AN INTERRUPT
002452  004767  006616                  JSR     %7,INIT         ;INITIALIZE
002456  012710  002530                  MOV     #TINT11,@ADINT  ;LOAD RETURN POINTER
002462  052767  000340  175306          BIS     #340,PSR        ;SET PROCESSOR STATUS TO LEVEL 7
002470  016760  175302  000002          MOV     PSR,2(ADINT)    ;LOAD RETURN PROCESSOR STATUS
002476  042767  000340  175272          BIC     #340,PSR        ;SET PROCESSOR PRIORITY TO 0
002504  012713  000101                  MOV     #101,@CRS       ;SET READ AND INTERRUPT ENABLE
002510  105713                          TSTB    @CRS            ;WAIT FOR COLUMN READY
002512  100376                          BPL     .-2
002514  016067  000002  175254          MOV     2(ADINT),PSR    ;RESTORE PROCESSOR TO HIGHEST PRIORITY
002522  005013                          CLR     @CRS            ;CLEAR INTERRUPT ENABLE
002524  104000                          HLT                     ;COLUMN READY DID NOT INTERRUPT
002526  000435                          BR      CONT11
002530  005013                  TINT11: CLR     @CRS            ;CLEAR INTERRUPT ENABLE
002532  105713                          TSTB    @CRS            ;MAKE SURE COLUMN READY IS SET
002534  100401                          BMI     .+4             ;BRANCH IF SET
002536  104000                          HLT                     ;COLUMN READY WASN'T SET
002540  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
002542  012710  000232          CONT11: MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
002546  005037  000232                  CLR     @#232           ;TO CAUSE A HALT IF ANOTHER INTERRUPT OCCURS

002552  104420                  TEST12: SCOPE
                                ;CARD DONE SHOULDN'T CAUSE AN INTERRUPT IF THE PROCESSOR IS AT LEVEL 7 PRIORITY
002554  004767  006514                  JSR     %7,INIT         ;INITIALIZE
002560  012710  002614                  MOV     #TINT12,@ADINT  ;SETUP RETURN
002564  052767  000340  175204          BIS     #340,PSR        ;SET PROCESSOR TO LEVEL 7 PRIORITY
002572  016760  175200  000002          MOV     PSR,2(ADINT)    ;LOAD RETURN PROCESSOR STATUS
002600  012713  000103                  MOV     #103,@CRS       ;SET EJECT, INTERRUPT ENABLE, AND READ
002604  032713  040000                  BIT     #40000,@CRS     ;WAIT FOR CARD DONE
002610  001775                          BEQ     .-4
002612  000402                          BR      .+6             ;CONTINUE IF NO INTERRUPT OCCURRED
002614  104000                  TINT12: HLT                     ;AN INTERRUPT OCCURRED
002616  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
002620  005013                          CLR     @CRS            ;CLEAR INTERRUPT ENABLE AND EJECT
002622  012710  000232                  MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
002626  005037  000232                  CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
```

```
                        ;FIND THE LEVEL AT WHICH AN INTERRUPT OCCURS
                        ;PRINT OUT A MESSAGE STATING THIS LEVEL IF IT IS OTHER THAN THE STANDARD (LEVEL 6)
                        ;MAKE CERTAIN THAT IT ALWAYS OCCURS AT THIS LEVEL
                        ;THE MESSAGE STATING THE LEVEL IS PRINTED ONLY ONCE, AND THE PROGRAM MUST
                        ;BE STARTED OVER AT LOCATION 200 FOR IT TO BE PRINTED AGAIN

                                .MACR   INT     NUM,PRP,STTS,INTLVL,NXTST
                        ;TEST FOR AN INTERRUPT ON LEVEL 'INTLVL
                        TEST'NUM:       SCOPE
                                JSR     %7,INIT         ;INITIALIZE
                                MOV     #TINT'NUM,@ADINT        ;SETUP RETURN ADDRESS
                                BIS     #340,PSR        ;SET PROCESSOR PRIORITY TO 7
                                MOV     PSR,2(ADINT)    ;SETUP RETURN PROCESSOR STATUS
                                BIC     #340,PSR        ;SET PROCESSOR PRIORITY TO 0
                                BIS     #STTS,PSR       ;SET PROCESSOR TO LEVEL PRP PRIORITY
                                MOV     #103,@CRS       ;SET EJECT INTERRUPT ENABLE, AND READ
                                BIT     #40000,@CRS     ;WAIT FOR CARD DONE
                                BEQ     .-4
                                MOV     2(ADINT),PSR    ;RESTORE PROCESSOR TO HIGHEST PRIORITY
                                CLR     @CRS            ;DISABLE INTERRUPTS
                                MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
                                CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
                                TST     INTFLG          ;CHECK TO SEE IF LEVEL ALREADY RECORDED
                                BPL     NXTST           ;IF NO, GO TO NEXT TEST
                                CMP     INTFLG,#10000'INTLVL    ;IF SO, CHECK TO SEE
                                BMI     NXTST           ;THAT THE INTERRUPT LEVEL RECORDED
                                                        ;IS BELOW THE CURRENT LEVEL
                                HLT                     ;INTERRUPT DIDN'T OCCUR WITH STATUS
                                                        ;AT LEVEL INTLVL, BUT PREVIOUSLY OCCURRED
                                                        ;AT OR ABOVE THIS LEVEL
                                BR      NXTST
                        TINT'NUM:       BIT     #40000,@CRS     ;MAKE SURE CARD DONE IS SET
                                BNE     .+4             ;BRANCH IF SET
                                HLT                     ;CARD DONE WASN'T SET
                                CLR     @CRS            ;DISABLE FURTHER INTERRUPTS
                                MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
                                CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
                                CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
                                TST     INTFLG          ;CHECK FOR PREVIOUS FLAG
                                BMI     SET'INTLVL      ;BRANCH IF FLAG SET
                                MOV     #10000'INTLVL,INTFLG ;SET FLAG AND LEVEL
                                MOV     #MSG4,R2        ;SETUP FOR PRINTOUT
                                JSR     %7,TOUT         ;PRINT MESSAGE "THE INTERRUPT LEVEL WAS"
                                MOV     #INTLVL,R2
                                JSR     %7,PROCT        ;PRINT LEVEL NUMBER
                                BR      NXTST
                        SET'INTLVL: CMP INTFLG,#10000'INTLVL    ;CHECK PREVIOUS LEVEL
                                BPL     NXTST
                                HLT                     ;INTERRUPT PREVIOUSLY OCCURRED ONLY AT A LOWER LEVEL
                                .ENDM

                                INT     13,6,300,7,TEST14
                        ;TEST FOR AN INTERRUPT ON LEVEL 7
002632  104400          TEST13: SCOPE
002634  004767  006434            JSR     %7,INIT         ;INITIALIZE
```

```
002640  012710  002750                MOV     #TINT13,@ADINT  ;SETUP RETURN ADDRESS
002644  052767  000340  175124        BIS     #340,PSR        ;SET PROCESSOR PRIORITY TO 7
002652  016760  175120  000002        MOV     PSR,2(ADINT)    ;SETUP RETURN PROCESSOR STATUS
002660  042767  000340  175110        BIC     #340,PSR        ;SET PROCESSOR PRIORITY TO 0
002666  052767  000300  175102        BIS     #300,PSR        ;SET PROCESSOR TO LEVEL 6 PRIORITY
002674  012713  000103                MOV     #103,@CRS       ;SET EJECT INTERRUPT ENABLE, AND READ
002700  032713  040000                BIT     #40000,@CRS     ;WAIT FOR CARD DONE
002704  001775                        BEQ     .-4
002706  016067  000002  175062        MOV     2(ADINT),PSR    ;RESTORE PROCESSOR TO HIGHEST PRIORITY
002714  005013                        CLR     @CRS            ;DISABLE INTERRUPTS
002716  012710  000232                MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
002722  005037  000232                CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
002726  005767  175650                TST     INTFLG          ;CHECK TO SEE IF LEVEL ALREADY RECORDED
002732  100044                        BPL     TEST14          ;IF NO, GO TO NEXT TEST
002734  026727  175642  100007        CMP     INTFLG,#100007  ;IF SO, CHECK TO SEE
002742  100440                        BMI     TEST14          ;THAT THE INTERRUPT LEVEL RECORDED
                                                              ;IS BELOW THE CURRENT LEVEL
002744  104020                        HLT                     ;INTERRUPT DIDN'T OCCUR WITH STATUS
                                                              ;AT LEVEL 7, BUT PREVIOUSLY OCCURRED
                                                              ;AT OR ABOVE THIS LEVEL
002746  000436                        BR      TEST14
002750  032713  040000        TINT13: BIT     #40000,@CRS     ;MAKE SURE CARD DONE IS SET
002754  001001                        BNE     .+4             ;BRANCH IF SET
002756  104000                        HLT                     ;CARD DONE WASN'T SET
002760  005013                        CLR     @CRS            ;DISABLE FURTHER INTERRUPTS
002762  012710  000232                MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
002766  005037  000232                CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
002772  022626                        CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
002774  005767  175602                TST     INTFLG          ;CHECK FOR PREVIOUS FLAG
003000  100414                        BMI     SET7    ;BRANCH IF FLAG SET
003002  012767  100007  175572        MOV     #100007,INTFLG  ;SET FLAG AND LEVEL
003010  012702  013627                MOV     #MSG4,R2        ;SETUP FOR PRINTOUT
003014  004767  006764                JSR     %7,TOUT         ;PRINT MESSAGE "THE INTERRUPT LEVEL WAS"
003020  012702  000007                MOV     #7,R2
003024  004767  006540                JSR     %7,PROCT        ;PRINT LEVEL NUMBER
003030  000425                        BR      TEST14
003032  026727  175544  100007  SET7: CMP     INTFLG,#100007  ;CHECK PREVIOUS LEVEL
003040  100001                        BPL     TEST14
003042  104020                        HLT                     ;INTERRUPT PREVIOUSLY OCCURRED ONLY AT A LOWER LEVEL


                                ;TEST FOR AN INTERRUPT ON LEVEL 6
                                ;SINCE THIS IS WHERE THE CARD READER NORMALLY IS, DON'T PRINT OUT A MESSAGE
                                ;IF IT IS FOUND HERE
003044  104420                TEST14: SCOPE
003046  004767  006222                JSR     %7,INIT         ;INITIALIZE
003052  012710  003142                MOV     #TINT14,@ADINT  ;SETUP RETURN ADDRESS
003056  052767  000340  174712        BIS     #340,PSR        ;SET PROCESSOR PRIORITY TO 7
003064  016760  174706  000002        MOV     PSR,2(ADINT)    ;SETUP RETURN PROCESSOR STATUS
003072  042767  000340  174676        BIC     #340,PSR        ;SET PROCESSOR PRIORITY TO 0
003100  052767  000240  174670        BIS     #240,PSR        ;SET PROCESSOR TO LEVEL 5 PRIORITY
003106  012713  000103                MOV     #103,@CRS       ;SET EJECT, INTERRUPT ENABLE, AND READ
003112  032713  040000                BIT     #40000,@CRS     ;WAIT FOR CARD DONE
003116  001775                        BEQ     .-4
003120  016067  000002  174650        MOV     2(ADINT),PSR    ;RESTORE PROCESSOR TO HIGHEST PRIORITY
```

```
003126  005013                          CLR     @CRS            ;DISABLE INTERRUPTS
003130  012710  000232                  MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
003134  005037  000232                  CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
003140  000426                          BR      TEST15
003142  032713  040000          TINT14: BIT     #40000,@CRS     ;MAKE SURE CARD DONE IS SET
003146  001001                          BNE     .+4             ;BRANCH IF SET
003150  104000                          HLT                     ;CARD DONE WASN'T SET
003152  005013                          CLR     @CRS            ;DISABLE FURTHER INTERRUPTS
003154  012710  000232                  MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
003160  005037  000232                  CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
003164  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
003166  005767  175410                  TST     INTFLG          ;CHECK FOR PREVIOUS FLAG
003172  100424                          BMI     SET14           ;BRANCH IF FLAG SET
003174  012767  100006  175400          MOV     #100006,INTFLG  ;SET FLAG AND LEVEL
003202  000405                          BR      TEST15
003204  026727  175372  100006  SET14: CMP     INTFLG,#100006  ;CHECK PREVIOUS LEVEL
003212  100001                          BPL     TEST15
003214  104000                          HLT                     ;INTERRUPT PREVIOUSLY OCCURRED ONLY AT A LOWER LEVEL

                                        INT     15,4,200,5,TEST16
                                ;TEST FOR AN INTERRUPT ON LEVEL 5
003216  104400                  TEST15: SCOPE
003220  004767  006050                  JSR     %7,INIT         ;INITIALIZE
003224  012710  003334                  MOV     #TINT15,@ADINT  ;SETUP RETURN ADDRESS
003230  052767  000340  174540          BIS     #340,PSR        ;SET PROCESSOR PRIORITY TO 7
003236  016760  174534  000002          MOV     PSR,2(ADINT)    ;SETUP RETURN PROCESSOR STATUS
003244  042767  000340  174524          BIC     #340,PSR        ;SET PROCESSOR PRIORITY TO 0
003252  052767  000200  174516          BIS     #200,PSR        ;SET PROCESSOR TO LEVEL 4 PRIORITY
003260  012713  000103                  MOV     #103,@CRS       ;SET EJECT INTERRUPT ENABLE, AND READ
003264  032713  040000                  BIT     #40000,@CRS     ;WAIT FOR CARD DONE
003270  001775                          BEQ     .-4
003272  016067  000002  174476          MOV     2(ADINT),PSR    ;RESTORE PROCESSOR TO HIGHEST PRIORITY
003300  005013                          CLR     @CRS            ;DISABLE INTERRUPTS
003302  012710  000232                  MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
003306  005037  000232                  CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
003312  005767  175264                  TST     INTFLG          ;CHECK TO SEE IF LEVEL ALREADY RECORDED
003316  100044                          BPL     TEST16          ;IF NO, GO TO NEXT TEST
003320  026727  175256  100005          CMP     INTFLG,#100005  ;IF SO, CHECK TO SEE
003326  100440                          BMI     TEST16          ;THAT THE INTERRUPT LEVEL RECORDED
                                                                ;IS BELOW THE CURRENT LEVEL
003330  104000                          HLT                     ;INTERRUPT DIDN'T OCCUR WITH STATUS
                                                                ;AT LEVEL 5, BUT PREVIOUSLY OCCURRED
                                                                ;AT OR ABOVE THIS LEVEL
003332  000436                          BR      TEST16
003334  032713  040000          TINT15: BIT     #40000,@CRS     ;MAKE SURE CARD DONE IS SET
003340  001001                          BNE     .+4             ;BRANCH IF SET
003342  104000                          HLT                     ;CARD DONE WASN'T SET
003344  005013                          CLR     @CRS            ;DISABLE FURTHER INTERRUPTS
003346  012710  000232                  MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
003352  005037  000232                  CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
003356  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
003360  005767  175216                  TST     INTFLG          ;CHECK FOR PREVIOUS FLAG
003364  100414                          BMI     SET5    ;BRANCH IF FLAG SET
003366  012767  100005  175206          MOV     #100005,INTFLG  ;SET FLAG AND LEVEL
003374  012722  013627                  MOV     #MSG4,R2        ;SETUP FOR PRINTOUT
```

```
003400  004767  006400                JSR     %7,TOUT         ;PRINT MESSAGE "THE INTERRUPT LEVEL WAS"
003404  012702  000005                MOV     #5,R2
003410  004767  006154                JSR     %7,PROCT        ;PRINT LEVEL NUMBER
003414  000435                       BR      TEST16
003416  026727  175160  100005  SET5: CMP     INTFLG,#100005  ;CHECK PREVIOUS LEVEL
003424  100001                       BPL     TEST16
003426  104000                       HLT                     ;INTERRUPT PREVIOUSLY OCCURRED ONLY AT A LOWER LEVEL

                                     INT     16,3,140,4,TEST17
                                ;TEST FOR AN INTERRUPT ON LEVEL 4
003430  104400                TEST16: SCOPE
003432  004767  005636                JSR     %7,INIT         ;INITIALIZE
003436  012710  003546                MOV     #TINT16,@ADINT  ;SETUP RETURN ADDRESS
003442  052767  000340  174326        BIS     #340,PSR        ;SET PROCESSOR PRIORITY TO 7
003450  016760  174322  000002        MOV     PSR,2(ADINT)    ;SETUP RETURN PROCESSOR STATUS
003456  042767  000340  174312        BIC     #340,PSR        ;SET PROCESSOR PRIORITY TO 0
003464  052767  000140  174304        BIS     #140,PSR        ;SET PROCESSOR TO LEVEL 3 PRIORITY
003472  012713  000103                MOV     #103,@CRS       ;SET EJECT INTERRUPT ENABLE, AND READ
003476  032713  040000                BIT     #40000,@CRS     ;WAIT FOR CARD DONE
003502  001775                       BEQ     .-4
003504  016067  000002  174264        MOV     2(ADINT),PSR    ;RESTORE PROCESSOR TO HIGHEST PRIORITY
003512  005013                       CLR     @CRS            ;DISABLE INTERRUPTS
003514  012710  000232                MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
003520  005037  000232                CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
003524  005767  175052                TST     INTFLG          ;CHECK TO SEE IF LEVEL ALREADY RECORDED
003530  100044                       BPL     TEST17          ;IF NO, GO TO NEXT TEST
003532  026727  175044  100004        CMP     INTFLG,#100004  ;IF SO, CHECK TO SEE
003540  100440                       BMI     TEST17          ;THAT THE INTERRUPT LEVEL RECORDED
                                                             ;IS BELOW THE CURRENT LEVEL
003542  104000                       HLT                     ;INTERRUPT DIDN'T OCCUR WITH STATUS
                                                             ;AT LEVEL 4, BUT PREVIOUSLY OCCURRED
                                                             ;AT OR ABOVE THIS LEVEL
003544  000436                       BR      TEST17
003546  032713  040000        TINT16: BIT     #40000,@CRS     ;MAKE SURE CARD DONE IS SET
003552  001001                       BNE     .+4             ;BRANCH IF SET
003554  104000                       HLT                     ;CARD DONE WASN'T SET
003556  005013                       CLR     @CRS            ;DISABLE FURTHER INTERRUPTS
003560  012710  000232                MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
003564  005037  000232                CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
003570  022626                       CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
003572  005767  175004                TST     INTFLG          ;CHECK FOR PREVIOUS FLAG
003576  100414                       BMI     SET4    ;BRANCH IF FLAG SET
003600  012767  100004  174774        MOV     #100004,INTFLG ;SET FLAG AND LEVEL
003606  012702  013627                MOV     #MSG4,R2        ;SETUP FOR PRINTOUT
003612  004767  006166                JSR     %7,TOUT         ;PRINT MESSAGE "THE INTERRUPT LEVEL WAS"
003616  012702  000004                MOV     #4,R2
003622  004767  005742                JSR     %7,PROCT        ;PRINT LEVEL NUMBER
003626  000435                       BR      TEST17
003630  026727  174746  100004  SET4: CMP     INTFLG,#100004  ;CHECK PREVIOUS LEVEL
003636  100001                       BPL     TEST17
003640  104000                       HLT                     ;INTERRUPT PREVIOUSLY OCCURRED ONLY AT A LOWER LEVEL

                                     INT     17,2,100,3,TEST18
                                ;TEST FOR AN INTERRUPT ON LEVEL 3
003642  104400                TEST17: SCOPE
```

```
003644  004767  005424                    JSR     %7,INIT          ;INITIALIZE
003650  012710  003760                    MOV     #TINT17,@ADINT   ;SETUP RETURN ADDRESS
003654  052767  000340  174114            BIS     #340,PSR         ;SET PROCESSOR PRIORITY TO 7
003662  016760  174110  000002            MOV     PSR,2(ADINT)     ;SETUP RETURN PROCESSOR STATUS
003670  042767  000340  174100            BIC     #340,PSR         ;SET PROCESSOR PRIORITY TO 0
003676  052767  000100  174072            BIS     #100,PSR         ;SET PROCESSOR TO LEVEL 2 PRIORITY
003704  012713  000103                    MOV     #103,@CRS        ;SET EJECT INTERRUPT ENABLE, AND READ
003710  032713  040000                    BIT     #40000,@CRS      ;WAIT FOR CARD DONE
003714  001775                            BEQ     .-4
003716  016067  000002  174052            MOV     2(ADINT),PSR     ;RESTORE PROCESSOR TO HIGHEST PRIORITY
003724  005013                            CLR     @CRS             ;DISABLE INTERRUPTS
003726  012710  000232                    MOV     #232,@ADINT      ;CHANGE INTERRUPT RETURN ADDRESS
003732  005037  000232                    CLR     @#232            ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
003736  005767  174640                    TST     INTFLG           ;CHECK TO SEE IF LEVEL ALREADY RECORDED
003742  100044                            BPL     TEST18           ;IF NO, GO TO NEXT TEST
003744  026727  174632  100003            CMP     INTFLG,#100003   ;IF SO, CHECK TO SEE
003752  100440                            BMI     TEST18           ;THAT THE INTERRUPT LEVEL RECORDED
                                                                   ;IS BELOW THE CURRENT LEVEL
003754  104020                            HLT                      ;INTERRUPT DIDN'T OCCUR WITH STATUS
                                                                   ;AT LEVEL 3, BUT PREVIOUSLY OCCURRED
                                                                   ;AT OR ABOVE THIS LEVEL
003756  000436                            BR      TEST18
003760  032713  040000          TINT17:   BIT     #40000,@CRS      ;MAKE SURE CARD DONE IS SET
003764  001001                            BNE     .+4              ;BRANCH IF SET
003766  104020                            HLT                      ;CARD DONE WASN'T SET
003770  005013                            CLR     @CRS             ;DISABLE FURTHER INTERRUPTS
003772  012710  000232                    MOV     #232,@ADINT      ;CHANGE INTERRUPT RETURN ADDRESS
003776  005037  000232                    CLR     @#232            ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
004032  022626                            CMP     (SP)+,(SP)+      ;RESTORE STACK POINTER
004004  005767  174572                    TST     INTFLG           ;CHECK FOR PREVIOUS FLAG
004010  100414                            BMI     SET3     ;BRANCH IF FLAG SET
004012  012767  100003  174562            MOV     #100003,INTFLG ;SET FLAG AND LEVEL
004020  012702  013627                    MOV     #MSG4,R2         ;SETUP FOR PRINTOUT
004024  004767  005754                    JSR     %7,TOUT          ;PRINT MESSAGE "THE INTERRUPT LEVEL WAS"
004030  012702  000003                    MOV     #3,R2
004034  004767  005530                    JSR     %7,PROCT         ;PRINT LEVEL NUMBER
004040  000425                            BR      TEST18
004042  026727  174534  100003  SET3:    CMP     INTFLG,#100003   ;CHECK PREVIOUS LEVEL
004050  100001                            BPL     TEST18
004052  104000                            HLT                      ;INTERRUPT PREVIOUSLY OCCURRED ONLY AT A LOWER LEVEL

                                          INT     18,1,040,2,TEST19
                                  ;TEST FOR AN INTERRUPT ON LEVEL 2
004054  104400                  TEST18: SCOPE
004056  004767  005212                    JSR     %7,INIT          ;INITIALIZE
004062  012710  004172                    MOV     #TINT18,@ADINT   ;SETUP RETURN ADDRESS
004066  052767  000340  173702            BIS     #340,PSR         ;SET PROCESSOR PRIORITY TO 7
004074  016760  173676  000002            MOV     PSR,2(ADINT)     ;SETUP RETURN PROCESSOR STATUS
004102  042767  000340  173666            BIC     #340,PSR         ;SET PROCESSOR PRIORITY TO 0
004110  052767  000040  173660            BIS     #040,PSR         ;SET PROCESSOR TO LEVEL 1 PRIORITY
004116  012713  000103                    MOV     #103,@CRS        ;SET EJECT INTERRUPT ENABLE, AND READ
004122  032713  040000                    BIT     #40000,@CRS      ;WAIT FOR CARD DONE
004126  001775                            BEQ     .-4
004130  016067  000002  173640            MOV     2(ADINT),PSR     ;RESTORE PROCESSOR TO HIGHEST PRIORITY
004136  005013                            CLR     @CRS             ;DISABLE INTERRUPTS
```

```
004140  012710  000232          MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
004144  005037  000232          CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
004150  005767  174426          TST     INTFLG          ;CHECK TO SEE IF LEVEL ALREADY RECORDED
004154  100044                  BPL     TEST19          ;IF NO, GO TO NEXT TEST
004156  026727  174420  100002  CMP     INTFLG,#100002  ;IF SO, CHECK TO SEE
004164  100440                  BMI     TEST19          ;THAT THE INTERRUPT LEVEL RECORDED
                                                        ;IS BELOW THE CURRENT LEVEL
004166  104000                  HLT                     ;INTERRUPT DIDN'T OCCUR WITH STATUS
                                                        ;AT LEVEL 2, BUT PREVIOUSLY OCCURRED
                                                        ;AT OR ABOVE THIS LEVEL
004170  000436                  BR      TEST19
004172  032713  040000  TINT18: BIT     #40000,@CRS     ;MAKE SURE CARD DONE IS SET
004176  001001                  BNE     .+4             ;BRANCH IF SET
004200  104000                  HLT                     ;CARD DONE WASN'T SET
004202  005013                  CLR     @CRS            ;DISABLE FURTHER INTERRUPTS
004204  012710  000232          MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
004210  005037  000232          CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
004214  022626                  CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
004216  005767  174360          TST     INTFLG          ;CHECK FOR PREVIOUS FLAG
004222  100414                  BMI     SET2    ;BRANCH IF FLAG SET
004224  012767  100002  174350  MOV     #100002,INTFLG  ;SET FLAG AND LEVEL
004232  012702  013627          MOV     #MSG4,R2        ;SETUP FOR PRINTOUT
004236  004767  005542          JSR     %7,TOUT         ;PRINT MESSAGE "THE INTERRUPT LEVEL WAS"
004242  012702  000002          MOV     #2,R2
004246  004767  005316          JSR     %7,PROCT        ;PRINT LEVEL NUMBER
004252  000435                  BR      TEST19
004254  026727  174322  100002  SET2:  CMP     INTFLG,#100002  ;CHECK PREVIOUS LEVEL
004262  100001                  BPL     TEST19
004264  104000                  HLT                     ;INTERRUPT PREVIOUSLY OCCURRED ONLY AT A LOWER LEVEL

                                INT     19,0,000,1,TEST20
                        ;TEST FOR AN INTERRUPT ON LEVEL 1
004266  104400          TEST19: SCOPE
004270  004767  005000          JSR     %7,INIT         ;INITIALIZE
004274  012710  004404          MOV     #TINT19,@ADINT  ;SETUP RETURN ADDRESS
004300  052767  000340  173470  BIS     #340,PSR        ;SET PROCESSOR PRIORITY TO 7
004306  016760  173464  000002  MOV     PSR,2(ADINT)    ;SETUP RETURN PROCESSOR STATUS
004314  042767  000340  173454  BIC     #340,PSR        ;SET PROCESSOR PRIORITY TO 0
004322  052767  000000  173446  BIS     #000,PSR        ;SET PROCESSOR TO LEVEL 0 PRIORITY
004330  012713  000103          MOV     #103,@CRS       ;SET EJECT INTERRUPT ENABLE, AND READ
004334  032713  040000          BIT     #40000,@CRS     ;WAIT FOR CARD DONE
004340  001775                  BEQ     .-4
004342  016067  000002  173426  MOV     2(ADINT),PSR    ;RESTORE PROCESSOR TO HIGHEST PRIORITY
004350  005013                  CLR     @CRS            ;DISABLE INTERRUPTS
004352  012710  000232          MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
004356  005037  000232          CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
004362  005767  174214          TST     INTFLG          ;CHECK TO SEE IF LEVEL ALREADY RECORDED
004366  100044                  BPL     TEST20          ;IF NO, GO TO NEXT TEST
004370  026727  174206  100001  CMP     INTFLG,#100001  ;IF SO, CHECK TO SEE
004376  100440                  BMI     TEST20          ;THAT THE INTERRUPT LEVEL RECORDED
                                                        ;IS BELOW THE CURRENT LEVEL
004400  104000                  HLT                     ;INTERRUPT DIDN'T OCCUR WITH STATUS
                                                        ;AT LEVEL 1, BUT PREVIOUSLY OCCURRED
                                                        ;AT OR ABOVE THIS LEVEL
004402  000436                  BR      TEST20
```

```
004404  032713  040000          TINT19: BIT     #40000,@CRS     ;MAKE SURE CARD DONE IS SET
004410  001001                          BNE     .+4             ;BRANCH IF SET
004412  104000                          HLT                     ;CARD DONE WASN'T SET
004414  005013                          CLR     @CRS            ;DISABLE FURTHER INTERRUPTS
004416  012710  000232                  MOV     #232,@ADINT     ;CHANGE INTERRUPT RETURN ADDRESS
004422  005037  000232                  CLR     @#232           ;TO CAUSE A HALT IF AN INTERRUPT OCCURS
004426  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
004430  005767  174146                  TST     INTFLG          ;CHECK FOR PREVIOUS FLAG
004434  100414                          BMI     SET1    ;BRANCH IF FLAG SET
004436  012767  100001  174136          MOV     #100001,INTFLG  ;SET FLAG AND LEVEL
004444  012702  013627                  MOV     #MSG4,R2        ;SETUP FOR PRINTOUT
004450  004767  005330                  JSR     %7,TOUT         ;PRINT MESSAGE "THE INTERRUPT LEVEL WAS"
004454  012702  000001                  MOV     #1,R2
004460  004767  005104                  JSR     %7,PROCT        ;PRINT LEVEL NUMBER
004464  000425                          BR      TEST20
004466  026727  174110  100001  SET1:  CMP     INTFLG,#100001  ;CHECK PREVIOUS LEVEL
004474  100001                          BPL     TEST20
004476  104000                          HLT                     ;INTERRUPT PREVIOUSLY OCCURRED ONLY AT A LOWER LEVEL

                                ;A TIMING ERROR SHOULDN'T CAUSE AN INTERRUPT
004500  104400                  TEST20: SCOPE
004502  004767  004566                  JSR     %7,INIT         ;INITIALIZE
004506  012710  004560                  MOV     #TINT20,@ADINT  ;LOAD RETURN POINTER
004512  052767  000340  173256          BIS     #340,PSR        ;SET PROCESSOR TO HIGHEST PRIORITY
004520  016760  173252  000002          MOV     PSR,2(ADINT)    ;LOAD RETURN PROCESSOR STATUS
004526  012713  000101                  MOV     #101,@CRS       ;SET INTERRUPT ENABLE AND READ
004532  032713  004000                  BIT     #4000,@CRS      ;WAIT FOR TIMING ERROR TO SET
004536  001775                          BEQ     .-4
004540  042767  000340  173230          BIC     #340,PSR        ;MOVE PROCESSOR TO LOWEST PRIORITY
004546  000240                          NOP                     ;CLOCK INTERRUPT IF IT OCCURRED
004550  016067  000002  173220          MOV     2(ADINT),PSR    ;MOVE PROCESSOR BACK TO HIGHEST PRIORITY
004556  000402                          BR      .+6
004560  104000                  TINT20: HLT                     ;TIMING ERROR CAUSED AN INTERRUPT
004562  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
004564  012710  000232                  MOV     #232,@ADINT     ;CHANGE INTERRUPT ADDRESS TO CAUSE A
004570  005037  000232                  CLR     @#232           ;HALT IF AN INTERRUPT OCCURS
004574  032713  040000                  BIT     #40000,@CRS     ;WAIT FOR CARD DONE
004600  001775                          BEQ     .-4
004602  005013                          CLR     @CRS            ;CLEAR INTERRUPT ENABLE

004604  104400                  TEST21: SCOPE
                                ;TEST FOR NO INTERRUPT OCCURING WITH INTERRUPT ENABLE SET AND REST CLEARED
004606  004767  004462                  JSR     %7,INIT         ;INITIALIZE CSR TO ZERO
004612  012710  004662                  MOV     #TNINT,@ADINT   ;SETUP RETURN ADDRESS
004616  052767  000340  173152          BIS     #340,PSR        ;SET PROCESSOR TO LEVEL 7
004624  016760  173146  000002          MOV     PSR,2(ADINT)    ;STORE PROCESSOR STATUS
004632  005067  173140                  CLR     PSR             ;SET PROCESSOR TO LEVEL 0
004636  012713  000100                  MOV     #100,@CRS       ;ENABLE INTERRUPTS
004642  005227  000000                  INC     #0              ;WAIT AWHILE
004646  001375                          BNE     .-4
004652  016067  000002  173120          MOV     2(ADINT),PSR    ;RESTORE PROCESSOR TO LEVEL 7
004656  005013                          CLR     @CRS            ;DISABLE FURTHER INTERRUPTS
004660  000403                          BR      CONT21
004662  104000                  TNINT:  HLT                     ;AN INTERRUPT OCCURRED
004664  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
```

```
004666  005013                          CLR     @CRS            ;DISABLE FURTHER INTERRUPTS
004670  005037  000232          CONT21: CLR     @#232           ;CHANGE INTERRUPT RETURN ADDRESS TO
004674  012710  000232                  MOV     #232,@ADINT     ;CAUSE A HALT IF AN INTERRUPT OCCURS

004700  104400                  TEST22: SCOPE
                                ;CHECK FOR SIMULTANEOUS INTERRUPTS ON MORE THAN ONE LEVEL
004702  004767  004366                  JSR     %7,INIT         ;INITIALIZE CSR TO ZERO
004706  012710  004744                  MOV     #T2INT,@ADINT   ;SETUP RETURN ADDRESS
004712  052767  000340  173056          BIS     #340,PSR        ;SET PROCESSOR TO LEVEL 7
004720  016760  173052  000002          MOV     PSR,2(ADINT)    ;STORE PROCESSOR STATUS
004726  042767  000340  173042          BIC     #340,PSR        ;SET PROCESSOR TO LEVEL 7
004734  012713  000103                  MOV     #103,@CRS       ;SET INTERRUPT ENABLE AND EJECT A CARD
004740  000001                          WAIT                    ;WAIT FOR INTERRUPT
004742  000776                          BR      .-2             ;SIT IF TRACE BIT IS SET
004744  022626                  T2INT:  CMP     (6)+,(6)+       ;RESTORE STACK POINTER
004746  012710  004770                  MOV     #T2INTA,@ADINT  ;CHANGE RETRUN ADDRESS
004752  005067  173020                  CLR     PSR             ;SET PROCESSOR TO LEVEL 0
004756  000240                          NOP                     ;WAIT
004760  016067  000002  173010          MOV     2(ADINT),PSR    ;RESTORE PROCESSOR TO LEVEL 7
004766  000402                          BR      CONT22
004770  022626                  T2INTA: CMP     (6)+,(6)+       ;RESTORE STACK
004772  104000                          HLT                     ;THE INTERRUPT OCCURRED AT 2 LEVELS
004774  005013                  CONT22: CLR     @CRS            ;DISABLE INTERRUPTS
004776  005037  000232                  CLR     @#232           ;CHANGE INTERRUPT RETURN ADDRESS TO
005002  012710  000232                  MOV     #232,@ADINT     ;CAUSE A HALT IF AN INTERRUPT OCCURS

005006  104400                  TEST23: SCOPE
                                ;ALL MODES OF ADDRESSING CRB1 OR CRB2 (DATO,DATOB,DATI) SHOULD CLEAR
                                ;COLUMN READY
005010  004767  004260                  JSR     %7,INIT         ;INITIALIZE
005014  005213                          INC     @CRS            ;START READING A CARD
005016  105713                          TSTB    @CRS            ;WAIT FOR COLUMN READY
005020  100376                          BPL     .-2
005022  005014                          CLR     @CRB1           ;DATO TO CRB1
005024  105713                          TSTB    @CRS            ;CHECK COLUMN READY
005026  100002                          BPL     CNT23A          ;BRANCH IF CLEARED
005030  104000                          HLT                     ;DATO TO CRB1 DIDN'T CLEAR READY
005032  000467                          BR      TEST24          ;GO TO NEXT TEST
005034  105713                  CNT23A: TSTB    @CRS            ;WAIT FOR COLUMN READY
005036  100376                          BPL     .-2
005040  105014                          CLRB    @CRB1           ;DATOB TO LOW BYTE OF CRB1
005042  105713                          TSTB    @CRS            ;CHECK COLUMN READY
005044  100002                          BPL     CNT23B          ;BRANCH IF CLEARED
005046  104000                          HLT                     ;DATOB TO CRB1 LOW BYTE DIDN'T CLEAR READY
005050  000460                          BR      TEST24          ;GO TO NEXT TEST
005052  105713                  CNT23B: TSTB    @CRS            ;WAIT FOR COLUMN READY
005054  100376                          BPL     .-2
005056  105064  000001                  CLRB    1(CRB1)         ;DATOB TO HIGH BYTE OF CRB1
005062  105713                          TSTB    @CRS            ;CHECK COLUMN READY
005064  100002                          BPL     CNT23C          ;BRANCH IF CLEARED
005066  104000                          HLT                     ;DATOB TO CRB1 HIGH BYTE DIDN'T CLEAR READY
005070  000450                          BR      TEST24          ;GO TO NEXT TEST
005072  105713                  CNT23C: TSTB    @CRS            ;WAIT FOR COLUMN READY
005074  100376                          BPL     .-2
005076  005714                          TST     @CRB1           ;DATI TO CRB1
```

```
005100  105713                    TSTB    @CRS            ;CHECK COLUMN READY
005102  100002                    BPL     CNT23D          ;BRANCH IF CLEARED
005104  104000                    HLT                     ;DATI TO CRB1 DIDN'T CLEAR READY
005106  000441                    BR      TEST24          ;GO TO NEXT TEST
005110  105713            CNT23D: TSTB    @CRS            ;WAIT FOR COLUMN READY
005112  100376                    BPL     .-2
005114  005077  173500            CLR     @CRB2           ;DATO TO CRB2
005120  105713                    TSTB    @CRS            ;CHECK COLUMN READY
005122  100002                    BPL     CNT23E          ;BRANCH IF CLEARED
005124  104000                    HLT                     ;DATO TO CRB2 DIDN'T CLEAR READY
005126  000431                    BR      TEST24          ;GO TO NEXT TEST
005130  105713            CNT23E: TSTB    @CRS            ;WAIT FOR COLUMN READY
005132  100376                    BPL     .-2
005134  105077  173460            CLRB    @CRB2           ;DATOB TO LOW BYTE OF CRB2
005140  105713                    TSTB    @CRS            ;CHECK COLUMN READY
005142  100002                    BPL     CNT23F          ;BRANCH IF CLEARED
005144  104000                    HLT                     ;DATOB TO CRB2 LOW BYTE DIDN'T CLEAR READY
005146  000421                    BR      TEST24          ;GO TO NEXT TEST
005150  105713            CNT23F: TSTB    @CRS            ;WAIT FOR COLUMN READY
005152  100376                    BPL     .-2
005154  016702  173440            MOV     CRB2,R2         ;LOAD POINTER
005160  105062  000001            CLRB    1(R2)           ;DATOB TO HIGH BYTE OF CRB2
005164  105713                    TSTB    @CRS            ;CHECK COLUMN READY
005166  100002                    BPL     CNT23G          ;BRANCH IF CLEARED
005170  104000                    HLT                     ;DATOB TO CRB2 HIGH BYTE DIDN'T CLEAR READY
005172  000407                    BR      TEST24          ;GO TO NEXT TEST

005174  105713            CNT23G: TSTB    @CRS            ;WAIT FOR COLUMN READY
005176  100376                    BPL     .-2
005200  005777  173414            TST     @CRB2           ;DATI TO CRB2
005204  105713                    TSTB    @CRS            ;CHECK COLUMN READY
005206  100001                    BPL     TEST24          ;BRANCH IF CLEARED
005210  104000                    HLT                     ;DATI TO CRB2 DIDN'T CLEAR READY

005212  104400            TEST24: SCOPE
                        ;SETTING EJECT AFTER A COLUMN READY WITHOUT CLEARING THE COLUMN READY
                        ;SHOULD SET TIMING ERROR (WHICH IN TURN SHOULD CLEAR COLUMN READY)
005214  004767  004054            JSR     %7,INIT         ;INITIALIZE
005220  005213                    INC     @CRS            ;START READING A CARD
005222  105713                    TSTB    @CRS            ;CHECK COLUMN READY - WAIT
005224  100376                    BPL     .-2
005226  052713  000002            BIS     #2,@CRS         ;SET EJECT
005232  105713                    TSTB    @CRS            ;CHECK COLUMN READY
005234  100402                    BMI     CNT24A          ;BRANCH IF STILL SET
005236  104000                    HLT                     ;SETTING EJECT CLEARED COLUMN READY
005240  000421                    BR      END24           ;BRANCH TO WAIT FOR DONE AFTER ERROR
005242  032713  004000    CNT24A: BIT     #4000,@CRS      ;CHECK TIMING ERROR
005246  001013                    BNE     TIM24           ;BRANCH IF SET
005250  032713  040400            BIT     #40400,@CRS     ;CHECK CARD DONE AND OFF-LINE
005254  001772                    BEQ     CNT24A          ;LOOP IF NONE SET
005256  032713  040000            BIT     #40000,@CRS     ;CARD DONE SET?
005262  001003                    BNE     CNT24B          ;YES - BRANCH TO ERROR PRINTOUT
005264  004767  004056            JSR     %7,CKBIT8       ;NO - BIT 8 WAS SET SO OUTPUT MESSAGE
005270  000415                    BR      ENDCK           ;BRANCH AFTER COMING BACK ON-LINE
005272  104000            CNT24B: HLT                     ;CARD DONE SET BUT TIMING ERROR DIDN'T
```

```
005274  000413                          BR      ENDCK           ;BRANCH TO NEXT SECTION
005276  105713                  TIM24:  TSTB    @CRS            ;CHECK COLUMN READY
005300  100001                          BPL     .+4             ;BRANCH IF NOT SET
005302  104000                          HLT                     ;TIMING ERROR DIDN'T CLEAR READY
005304  032713  040400          END24:  BIT     #40400,@CRS     ;WAIT FOR CARD DONE OR OFF-LINE
005310  001775                          BEQ     END24
005312  032713  000400                  BIT     #400,@CRS       ;CHECK OFF LINE
005316  001402                          BEQ     ENDCK           ;BRANCH IF NOT SET
005320  004767  004022                  JSR     %7,CKBIT8       ;OUTPUT ERROR MESSAGE

                                ;CHECK SW7 AND RETURN TO TEST1 IF SET, AFTER RINGING BELL
                                ;OTHERWISE GO INTO THE DATA TEST
005324  104400                  ENDCK:  SCOPE
005326  032767  000200  172234          BIT     #200,SR
005334  001406                          BEQ     DATST
005336  004767  003760                  JSR     %7,BELL
005342  005167  173256                  COM     TRFLG           ;TOGGLE TRACE FLAG
005346  000167  173330                  JMP     RESTRT


                                ;DATA RELIABILITY TEST FOR CR11

                                ;CHECK SR FOR TYPE OF DECK BEING TESTED, AND INITIALIZE POINTERS
005352  012767  000056  001172  DATST:  MOV     #56,CDCNT       ;SETUP CARD COUNT TO ENTER TABLE CORRESPONDING TO NEXT CARD
005360  000402                          BR      DATST2          ;SKIP NEXT INSTRUCTION
005362  005067  001164          DATST1: CLR     CDCNT           ;SETUP CARD COUNT TO ENTER DATA TABLE AT BEGINNING
005366  005067  173236          DATST2: CLR     ERFLG           ;FLAG SET PREVENTS PRINTING OUT ERROR HEADING
005372  032767  000020  172170          BIT     #20,SR          ;CHECK BIT 4 OF SR FOR TYPE OF DECK
005400  001412                          BEQ     ALP1            ;BRANCH IF NOT SET TO LOAD ALPHANUMERIC POINTERS
005402  012767  012650  001136          MOV     #BINCD,TSTART   ;BIT 2 SET, LOAD BINARY TABLE POINTERS
005410  012767  013346  001132          MOV     #BINEND,TEND
005416  012767  014753  001120          MOV     #MSG15,DECK
005424  000411                          BR      CONTD           ;BRANCH AROUND ALPHANUMERIC POINTERS
005426  012767  012150  001112  ALP1:   MOV     #ALPCD,TSTART   ;LOAD ALPHANUMERIC TABLE POINTERS
005434  012767  012646  001106          MOV     #ALPEND,TEND
005442  012767  014742  001074          MOV     #MSG14,DECK
005450  005767  173150          CONTD:  TST     TRFLG           ;CHECK TRACE TRAP FLAG
005454  001004                          BNE     TRP1            ;BRANCH IF FLAG WAS SET
005456  012767  000340  172312  NOTRP1: MOV     #340,PSR        ;CLEAR TRACE BIT
005464  000407                          BR      DCNT1
005466  032767  010000  172074  TRP1:   BIT     #10000,SR       ;CHECK SW12 TO INHIBIT TRACE TRAPPING
005474  001370                          BNE     NOTRP1          ;BRANCH IF SET
005476  012767  000360  172272          MOV     #360,PSR        ;SET TRACE BIT
005504  004767  003564          DCNT1:  JSR     %7,INIT         ;INITIALIZE CARD READER STATUS REGISTER
                                ;SET UP INTERRUPT SERVICING, AND START READING
005510  012710  005544                  MOV     #SRVC,@ADINT    ;SETUP RETURN POINTER
005514  042767  000340  172254          BIC     #340,PSR        ;SET PROCESSOR TO LEVEL 0
005522  016760  172250  000002          MOV     PSR,2(ADINT)    ;STORE CURRENT STATUS
005530  004767  000710                  JSR     %7,NXCRD        ;ADJUST POINTER AND START READING
005534  052713  000101                  BIS     #101,@CRS       ;ENABLE INTERRUPTS
005540  000001                          WAIT                    ;WAIT FOR INTERRUPTS
005542  000776                          BR      .-2

                                ;INTERRUPT SERVICE ROUTINE WHICH RUNS DATA RELIABILITY TEST
005544  005713                  SRVC:   TST     @CRS            ;CHECK SPECIAL CONDITION (BIT 15)
```

```
005546  100460                          BMI     ERSET           ;BRANCH IF SET
005550  105713                          TSTB    @CRS            ;CHECK COLUMN READY
005552  100402                          BMI     .+6             ;BRANCH IF SET
005554  000167  000536                  JMP     NOTCOL          ;JUMP IF NOT SET
005560  005267  000770                  INC     CLCNT           ;KEEP TRACK OF COLUMN NUMBER
005564  011467  000766                  MOV     @CRB1,DAT1      ;STORE DATA OF FIRST READ
005570  105713                          TSTB    @CRS            ;MAKE SURE COLUMN READY CLEARED
005572  100006                          BPL     SCONT1          ;BRANCH IF IT DID
005574  052767  000340  172174          BIS     #340,PSR        ;SET PROCESSOR TO LEVEL 7
005602  104000                          HLT                     ;READING DATA DIDN'T CLEAR COLUMN READY
005604  000167  000526                  JMP     LASTCK          ;GO TO NEXT CARD AFTER ERROR PRINTOUT
005610  017767  173004  000744  SCONT1: MOV     @CRB2,DATENC    ;STORE ENCODED DATA
005616  012701  000010                  MOV     #10,COUNT       ;WAIT AWHILE
005622  005301                          DEC     COUNT
005624  001376                          BNE     .-2
005626  011467  000726                  MOV     @CRB1,DAT2      ;STORE DATA OF SECOND READ
005632  005067  000726                  CLR     PTOFF           ;CLEAR POINTER OFFSET
005636  026715  000714                  CMP     DAT1,@R5        ;CHECK FIRST DATA READ
005642  001052                          BNE     FAIL            ;PRINTOUT IF WRONG
005644  012767  000002  000712          MOV     #2,PTOFF        ;SET POINTER OFFSET
005652  026725  000702                  CMP     DAT2,(R5)+      ;CHECK SECOND READING OF SAME DATA
005656  001044                          BNE     FAIL            ;BRANCH IF WRONG
005660  012767  000004  000676          MOV     #4,PTOFF        ;SET POINTER OFFSET
005666  026725  000670                  CMP     DATENC,(R5)+    ;CHECK ENCODED DATA
005672  001036                          BNE     FAIL            ;BRANCH IF WRONG
005674  020567  000650                  CMP     R5,TEND         ;CHECK FOR END OF TABLE
005700  100402                          BMI     .+6             ;IF NOT THERE, RTI
005702  016705  000640                  MOV     TSTART,R5       ;MOVE POINTER TO LOOP THRU TABLE
005706  000002                          RTI

                                        ;SPECIAL CONDITION BIT 15 WAS SET WHEN THE INTERRUPT SERVICE ROUTINE
                                        ;WAS ENTERED
                                        ;OUTPUT A MESSAGE AND HALT
005710  052767  000340  172060  ERSET;  BIS     #340,PSR        ;LOCK OUT INTERRUPTS
005716  022767  000120  000626          CMP     #80.,CDCNT      ;CHECK FOR LAST CARD
005724  001006                          BNE     ER1             ;IF NOT, PRINT OUT MESSAGE
005726  022767  000120  000620          CMP     #80.,CLCNT      ;IF LAST CARD, CHECK FOR LAST COLUMN
005734  001002                          BNE     ER1             ;IF NOT, PRINT MESSAGE
005736  000167  000624                  JMP     ALLDON          ;IF END OF DECK, JUMP
005742  012702  014764          ER1:    MOV     #MSG16,R2       ;"BIT 15 WAS SET,"
005746  004767  004032                  JSR     %7,TOUT
005752  012702  015005                  MOV     #MSG17,R2       ;"REMEDY THE ERROR CONDITION
005756  004767  004022                  JSR     %7,TOUT         ;AND PRESS CONTINUE"
005762  000000                          HALT
005764  000167  000346                  JMP     LASTCK          ;SET UP FOR NEXT CARD AND GO ON
005770  052767  000340  172000  FAIL:   BIS     #340,PSR        ;LOCK OUT INTERRUPTS
005776  052713  000002                  BIS     #2,@CRS         ;SET EJECT TO PREVENT TIMING ERROR
006002  005714                          TST     @CRB1           ;MAKE SURE COLUMN READY IS CLEARED
006004  032767  020000  171556          BIT     #20000,SR       ;CK SW13
006012  001431                          BEQ     FAILCN          ;CONTINUE IF NOT SET
006014  005767  171550                  TST     SR              ;IF SET, CHECK FOR HALT ON ERROR
006020  100003                          BPL     FAILC           ;BRANCH IF HALT ON ERROR NOT SET
006022  000000                          HALT                    ;HALT ON ERROR SET
006024  000167  000326                  JMP     LASTCK          ;CONTINUE  AFTER HALT
```

```
006030  032713  040000          FAILC:  BIT     #40000, @CRS    ;CHECK FOR CARD DONE
006034  001402                  BEQ     .+6
006036  000167  000274                  JMP     LASTCK          ;INHIBIT PRINTOUT AFTER CARD DONE SET
006042  032713  000400                  BIT     #400,@CRS       ;CHECK FOR OFF-LINE
006046  001770                  BEQ     FAILC           ;BRANCH IF NOT
006050  022767  000120  000474        CMP     #80.,CDCNT      ;CHECK FOR LAST CARD
006056  001002                  BNE     .+6
006060  000167  000502                  JMP     ALLDON          ;IF LAST CARD, WAIT FOR NEXT DECK
006064  004767  003256                  JSR     %7,CKBIT8       ;IF NOT LAST CARD, PRINT MESSAGE
006070  004767  000350                  JSR     %7,NXCRD        ;START NEXT CARD THRU READER
006074  000002                  RTI
006076  005767  172526          FAILCN: TST     ERFLG           ;TEST FLAG FOR PREVIOUS PRINTOUT
006102  001006                  BNE     NOHD            ;IF SET,  DON'T OUTPUT HEADING
006104  005267  172520                  INC     ERFLG           ;SET FLAG
006110  012702  014652                  MOV     #MSG13,R2       ;OUTPUT HEADING FOR DATA ERROR PRINTOUT
006114  004767  003664                  JSR     %7,TOUT
006120  016702  000420          NOHD:   MOV     DECK,R2         ;OUTPUT TYPE OF DECK
006124  004767  003654                  JSR     %7,TOUT
006130  004767  003246                  JSR     %7,SPACE
006134  016702  000412                  MOV     CDCNT,R2        ;OUTPUT CARD NUMBER WHERE ERROR OCCURRED
006140  004767  003424                  JSR     %7,PROCT
006144  004767  003232                  JSR     %7,SPACE
006150  016702  000400                  MOV     CLCNT,R2        ;OUTPUT COLUMN NUMBER WHERE ERROR OCCURRED
006154  004767  003410                  JSR     %7,PROCT
006160  004767  003216                  JSR     %7,SPACE
006164  166705  000374                  SUB     PTOFF,R5        ;SUBTRACT OFFSET FROM POINTER TO POINT TO
                                                                ;ADDRESS OF DESIRED PATTERN
006170  012502                  MOV     (R5)+,R2        ;OUTPUT CORRECT DATA PATTERN (NOT ENCODED)
006172  004767  003372                  JSR     %7,PROCT
006176  004767  003200                  JSR     %7,SPACE
006202  016702  000350                  MOV     DAT1,R2         ;OUTPUT DATA READ ON FIRST READING OF BUFFER
006206  004767  003356                  JSR     %7,PROCT
006212  004767  003164                  JSR     %7,SPACE
006216  016702  000336                  MOV     DAT2,R2         ;OUTPUT DATA READ ONE MILLISECOND LATER
006222  004767  003342                  JSR     %7,PROCT
006226  004767  003150                  JSR     %7,SPACE
006232  011502                  MOV     @R5,R2          ;OUTPUT CORRECT DATA PATTERN (ENCODED FORM)
006234  004767  003330                  JSR     %7,PROCT
006240  004767  003136                  JSR     %7,SPACE
006244  016702  000312                  MOV     DATENC,R2       ;OUTPUT DATA READ (ENCODED)
006250  004767  003314                  JSR     %7,PROCT
006254  005767  171310                  TST     SR              ;CHECK "HALT ON ERROR" SWITCH
006260  100001                  BPL     .+4             ;BRANCH IF NOT SET
006262  000000                  HALT                    ;HALT AFTER AN ERROR
006264  005713                  TST     @CRS            ;CHECK ERROR
006266  100023                  BPL     LASTCK          ;BRANCH IF NOT SET
006270  022767  000120  000254        CMP     #80.,CDCNT      ;CHECK FOR LAST CARD
006276  201005                  BNE     FAILC1
006300  032713  000400                  BIT     #400,@CRS
006304  001423                  BEQ     LASTCD
006306  000167  000254                  JMP     ALLDON
006312  000167  177372          FAILC1: JMP     ERSET           ;OUTPUT ERROR MESSAGE

                                ;INTERRUPT NOT DUE TO ERROR OR COLUMN READY
006316  032713  040000          NOTCOL: BIT     #40000,@CRS     ;CHECK FOR CARD DONE
```

```
006322  001474                          BEQ     NOTCD           ;BRANCH IF NOT SET
006324  022767  000120  000222          CMP     #80.,CLCNT      ;CHECK COLUMN COUNT
006332  001401                          BEQ     .+4             ;SKIP ERROR HALT IF 80 COLUMNS WERE READ
006334  104000                          HLT                     ;LESS THAN EIGHTY COLUMNS WERE READ
006336  022767  000120  000206  LASTCK: CMP     #80.,CDCNT      ;CHECK FOR LAST CARD
006344  001403                          BEQ     LASTCD          ;BRANCH IF LAST CARD
006346  004767  000072                  JSR     %7,NXCRD        ;IF NOT LAST CARD
006352  000002                          RTI                     ;GO ON
006354  022626                  LASTCD: CMP     (SP)+,(SP)+     ;IF LAST CARD, RESTORE STACK POINTER

006356  004767  002740                  JSR     %7,BELL         ;RING BELL TO SIGNIFY "PASS COMPLETE"
006362  013722  000042                  MOV     @#42,R2         ;MONITOR HOOK
006366  001405                          BEQ     END
006370  000005                          RESET
006372  004712                  LOGIC:  JSR     %7,(R2)
006374  000240                          NOP
006376  000240                          NOP
006400  000240                          NOP
006402  032767  000040  171160  END:    BIT     #40,SR          ;CHECK SR FOR CONTINUATION TO ANOTHER DECK
006410  001002                          BNE     .+6             ;BRANCH TO HALT IF SW5 SET
006412  000167  000002                  JMP     DECKCK          ;CONTINUE TO ANOTHER DECK
006416  000000                          HALT                    ;DATA TEST DONE

                                ;WHEN CONTINUING FROM ONE DECK TO ANOTHER, CHECK SW6 FOR TYPE
                                ;OF TESTING TO BE PERFORMED
006420  005167  172200          DECKCK: COM     TRFLG           ;TOGGLE TRACE FLAG
006424  032767  000100  171136          BIT     #100,SR         ;CHECK SW6
006432  001402                          BEQ     .+6             ;BRANCH IF NOT SET
006434  000167  172242                  JMP     RESTRT          ;RERUN COMBINED INSTRUCTION AND DATA TEST
006440  000167  176716                  JMP     DATST1

006444  016705  000076          NXCRD:  MOV     TSTART,R5       ;LOAD R5 WITH TABLE STARTING ADDRESS
006450  006367  000076                  ASL     CDCNT           ;MULTIPLY CARD COUNT BY FOUR
006454  006367  000072                  ASL     CDCNT
006460  066705  000066                  ADD     CDCNT,R5        ;ADD OFFSET TO R5 TO POINT TO NEXT DATUM
006464  006267  000062                  ASR     CDCNT           ;RESTORE CARD COUNT
006470  006267  000056                  ASR     CDCNT
006474  042713  000002                  BIC     #2,@CRS         ;CLEAR EJECT IF SET
006500  005213                          INC     @CRS            ;READ ANOTHER CARD
006502  005267  000044                  INC     CDCNT           ;KEEP TRACK OF CARD NUMBER
006506  005067  000042                  CLR     CLCNT           ;INITIALIZE COLUMN COUNT
006512  000207                          RTS     %7              ;RETURN

                                ;INTERRUPT NOT CAUSED BY ERROR, COLUMN READY, OR CARD DONE
006514  052767  000340  171254  NOTCD:  BIS     #340,PSR        ;LOCK OUT FURTHER INTERRUPTS
006522  032713  002000                  BIT     #2000,@CRS      ;TEST ON-LINE TRANSITION BIT
006526  001003                          BNE     NOTCD1          ;BRANCH IF SET
006530  104000                          HLT                     ;NO BITS SET TO CAUSE AN INTERRUPT
006532  000167  177600                  JMP     LASTCK          ;START NEXT CARD
006536  104000                  NOTCD1: HLT                     ;ON-LINE TRANSITION CAUSED AN INTERRUPT
006540  000167  177572                  JMP     LASTCK          ;START NEXT CARD
006544  000000                  DECK:   0                       ;POINTER TO LITERAL "ALPHA" OR "BINARY"
006546  000000                  TSTART: 0                       ;STARTING ADDRESS OF DATA TABLE
006550  000000                  TEND:   0                       ;END ADDRESS OF DATA TABLE
006552  000000                  CDCNT:  0                       ;NUMBER OF CARD BEING READ
```

```
006554  000000                  CLCNT:  0                       ;NUMBER OF COLUMN BEING CHECKED
006556  000000                  DAT1:   0                       ;DATA ON FIRST READ FROM CRB1
006560  000000                  DAT2:   0                       ;DATA ON SECOND READ OF CRB1
006562  000000                  DATENC: 0                       ;DATA READ FROM CRB2
006564  000000                  PTOFF:  0                       ;OFFSET TO POINTER FOR DATA PRINTOUT
006566  004767  002530          ALLDON: JSR     %7,BELL         ;RING BELL
006572  032713  000400                  BIT     #400,@CRS       ;CHECK OFF-LINE BIT
006576  001001                  BNE     .+4                     ;BRANCH IF SET
006600  104000                  HLT                             ;OFF-LINE NOT SET, BUT SPECIAL CONDITION
                                                                ;WAS SET AFTER 80 COLUMNS OF THE 80TH CARD WERE READ
006602  032767  000040  170760          BIT     #40,SR          ;CHECK SR FOR HALT AT END OF DECK
006610  001403                  BEQ     ALCNT                   ;CONTINUE IF NOT SET
006612  000000                  HALT                            ;END OF DECK,SW5 SET
006614  000167  177600                  JMP     DECKCK          ;CHECK FOR TYPE OF TESTING
006620  032767  002000  170742  ALCNT:  BIT     #2000,SR        ;DOES THIS CR11 USE THE M829 MODULE?
006626  001025                  BNE     ALCNT1                  ;YES- BRANCH
006630  005027  000000                  CLR     #0              ;NO-STALL TO ALLOW CARD DONE TO SET
006634  005367  177772                  DEC     .-2
006640  001375                  BNE     .-4
006642  005327  000000                  DEC     #0
006646  001375                  BNE     .-4
006650  005327  000000                  DEC     #0
006654  001375                  BNE     .-4
006656  032713  040000                  BIT     #40000, @CRS    ;CHECK CARD DONE
006662  001001                  BNE     .+4
006664  104000                  HLT                             ;CARD DONE DIDN'T SET- THIS ERROR COULD BE
006666  005013                  CLR     @CRS                    ;CAUSED BY RUNNING A CR11 WHICH HAS THE
                                                                ;M829 MODULE AND NOT SETTING SWITCH REGISTER
                                                                ;SWITCH 10

006670  032713  177377                  BIT     #177377,@CRS    ;ONLY BIT 8 MAY STILL BE SET
006674  001401                  BEQ     .+4                     ;BRANCH IF OK
006676  104000                  HLT                             ;STATUS REGISTER INCORRECT
006700  000405                  BR ALCNT2
006702  005013                  ALCNT1: CLR     @CRS            ;CLEAR ERROR
006704  032713  176377                  BIT     #176377,@CRS    ;ONLY BITS 8 AND 9 MAY STILL BE SET
                                                                ;BIT 9 MAY BE SET SINCE CARD MAY NOT
                                                                ;YET HAVE CLEARED THE READER TO CAUSE
                                                                ;CARD DONE
006710  001401                  BEQ     .+4
006712  104000                  HLT                             ;STATUS REGISTER INCORRECT
006714  052767  000340  171054  ALCNT2: BIS     #340,PSR        ;SET PROCESSOR TO LEVEL 7
006722  016760  171050  000002          MOV     PSR,2(ADINT)    ;SETUP RETURN STATUS
006730  105213                  INCB    @CRS                    ;ATTEMPT TO READ- SHOULD RESET ERROR
006732  005713                  TST     @CRS                    ;CHECK BIT 15
006734  100402                  BMI     ALLOK                   ;BRANCH IF OK
006736  104000                  HLT                             ;SETTING READ DIDN'T RESET ERROR
006740  000416                  BR      ALWAIT                  ;BRANCH TO WAIT FOR ON-LINE
006742  012710  006774  ALLOK:  MOV     #SRVC1,@ADINT           ;LOAD INTERRUPT RETURN ADDRESS
006746  005067  171024                  CLR     PSR             ;SET PROCESSOR TO LEVEL 0
006752  012713  000101                  MOV     #101,@CRS       ;ENABLE INTERRUPTS, KEEP ERROR SET BY SETTING READ
006756  000240                  NOP                             ;CLOCK IN INTERRUPT
006760  016067  000002  171010          MOV     2(ADINT),PSR    ;SET PROCESSOR TO LEVEL 7
006766  005013                  CLR     @CRS                    ;CLEAR INTERRUPT ENABLE AND ERROR
006770  104000                  HLT                             ;BIT 15 DIDN'T CAUSE AN INTERRUPT
```

```
006772  000402                          BR      .+6
006774  022626                  SRVC1:  CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
006776  005013                  ALWAIT: CLR     @CRS            ;CLEAR INTERRUPT ENABLE AND ERROR
007000  012710  007036                  MOV     #SRVC2,@ADINT   ;CHANGE INTERRUPT RETURN ADDRESS
007004  112713  000100                  MOVB    #100,@CRS       ;ENABLE INTERRUPTS
007010  042767  000340  170760          BIC     #340,PSR        ;SET PROCESSOR TO LEVEL 0
007016  032713  000400                  BIT     #400,@CRS       ;CHECK OFF-LINE BIT
007022  001375                          BNE     .-4             ;LOOP UNTIL CLEAR
007024  016067  000002  170744          MOV     2(ADINT),PSR    ;SET PROCESSOR TO LEVEL 7
007032  104000                          HLT                     ;NO INTERRUPT OCCURRED
007034  000403                          BR      SRVC2A          ;BRANCH AROUND
007036  004767  002260          SRVC2:  JSR     %7,BELL         ;RING BELL
007042  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
007044  032713  002000          SRVC2A: BIT     #2000,@CRS      ;CHECK BIT 10
007050  001001                          BNE     .+4             ;BRANCH IF SET
007052  104000                          HLT                     ;BIT 10 NOT SET
007054  032713  000400                  BIT     #400,@CRS       ;CHECK BIT 8
007060  001401                          BEQ     .+4             ;BRANCH IF NOT SET
007062  104000                          HLT                     ;BIT 8 WAS SET
007064  005013                          CLR     @CRS            ;DATO TO CRS
007066  032713  002000                  BIT     #2000,@CRS      ;CHECK BIT 10
007072  001401                          BEQ     .+4             ;BRANCH IF NOT SET
007074  104000                          HLT                     ;DATO DIDN'T CLEAR ON-LINE BIT
007076  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK FROM INITIAL INTERRUPT
007100  000167  177314                  JMP     DECKCK          ;RESTART

007104  005067  171502          ERCR11: CLR     FLAG
007110  000403                          BR      TSTA
007112  012767  000001  171472  ERCM11: MOV     #1,FLAG
007120  004767  171506          TSTA:   JSR     %7,SETUP        ;INITIALIZE REGISTERS
007124  012767  007134  002650          MOV     #TESTA+2,RETURN ;SETUP SCOPE LOOP RETURN ADDRESS
                        ;THE CARD READER GOING OFF-LINE SHOULD SET SPECIAL CONDITION (BIT 15) AND OFF-LINE (BIT 8)
007132  104400                  TESTA:  SCOPE
007134  005067  002636                  CLR     ITMAX           ;RUN EACH ERROR TEST ONCE ONLY
007140  004767  002130                  JSR     %7,INIT         ;INITIALIZE STATUS REGISTER
007144  012702  013534                  MOV     #MSG3,R2        ;"PRESS CARD READER 'READ STOP'"
007150  005767  171436                  TST     FLAG            ;CHANGE MESSAGE FOR DOCUMATION READER?
007154  001402                          BEQ     .+6             ;NO
007156  012702  013574                  MOV     #MSG3A,R2       ;"PRESS CARD READER 'STOP'"
007162  004767  002616                  JSR     %7,TOUT
007166  012702  013467                  MOV     #MSG2,R2
007172  004767  002606                  JSR     %7,TOUT         ;"THEN HIT 'CONTINUE' ON THE CONSOLE"
007176  004767  002724                  JSR     %7,CRLF4        ;MOVE MESSAGE UP ON TTY
007202  000000                          HALT
007204  032713  000400                  BIT     #400,@CRS       ;CHECK BIT 8
007210  001001                          BNE     .+4             ;BRANCH IF SET
007212  104000                          HLT                     ;OFF-LINE (BIT 8) WASN'T SET
007214  005713                          TST     @CRS            ;CHECK BIT 15
007216  100401                          BMI     .+4             ;BRANCH IF SET
007220  104000                          HLT                     ;BIT 15 WASN'T SET
007222  012702  013350                  MOV     #MSG1,R2        ;"PRESS CARD READER 'MOTOR START' AND 'READ START'";
007226  005767  171360                  TST     FLAG            ;CHANGE MESSAGE FOR DOCUMATION READER?
007232  001402                          BEQ     .+6             ;NO
007234  012702  013433                  MOV     #MSG1A,R2       ;"PRESS CARD READER 'RESET'"
007240  004767  002540                  JSR     %7,TOUT
```

```
007244  012702  013467                  MOV     #MSG2,R2
007250  004767  002530                  JSR     %7,TOUT         ;"THEN HIT 'CONTINUE' ON THE CONSOLE"
007254  004767  002646                  JSR     %7,CRLF4        ;MOVE MESSAGE UP ON TTY
007260  000000                          HALT
007262  032713  000400                  BIT     #400,@CRS       ;WAIT FOR OFF-LINE TO CLEAR
007266  001375                          BNE     .-4

                                ;INPUT HOPPER EMPTY SHOULD SET SPECIAL CONDITION
007270  104400                  TESTB:  SCOPE
007272  004767  001776                  JSR     %7,INIT         ;INITIALIZE STATUS REGISTER
007276  012702  013662                  MOV     #MSG5,R2        ;"REMOVE ALL CARDS FROM THE INPUT HOPPER"
007302  004767  002476                  JSR     %7,TOUT
007306  012702  013467                  MOV     #MSG2,R2        ;"THEN HIT 'CONTINUE' ON THE CONSOLE"
007312  004767  002466                  JSR     %7,TOUT
007316  004767  002604                  JSR     %7,CRLF4        ;MOVE MESSAGE UP ON TTY
007322  000000                          HALT
007324  032713  000400                  BIT     #400,@CRS       ;CHECK BIT8
007330  001001                          BNE     .+4             ;BRANCH IF SET
007332  104000                          HLT                     ;OFF-LINE (BIT 8) WASN'T SET
007334  005713                          TST     @CRS            ;CHECK SPECIAL CONDITION BIT
007336  100401                          BMI     .+4             ;BRANCH IF SET
007340  104000                          HLT                     ;SPECIAL CONDITION NOT SET
007342  012702  013733                  MOV     #MSG6,R2        ;"RESTORE CARDS IN INPUT HOPPER"
007346  004767  002432                  JSR     %7,TOUT
007352  012702  013350                  MOV     #MSG1,R2        ;"PRESS CARD READER 'MOTOR START' AND 'READ START'"
007356  005767  171230                  TST     FLAG            ;CHANGE MESSAGE FOR DOCUMATION READER?
007362  001402                          BEQ     .+6             ;NO
007364  012702  013433                  MOV     #MSG1A,R2       ;"PRESS CARD READER 'RESET'"
007370  004767  002410                  JSR     %7,TOUT
007374  012702  013467                  MOV     #MSG2,R2        ;"THEN HIT 'CONTINUE' ON THE CONSOLE"
007400  004767  002400                  JSR     %7,TOUT
007404  004767  002516                  JSR     %7,CRLF4        ;MOVE MESSAGE UP ON TTY
007410  000000                          HALT
007412  032713  000400                  BIT     #400,@CRS       ;WAIT FOR OFF-LINE TO CLEAR
007416  001375                          BNE     .-4

                                ;OUTPUT STACKER FULL SHOULD SET BIT 15
007420  104400                  TESTC:  SCOPE
007422  004767  001646                  JSR     %7,INIT         ;INITIALIZE STATUS REGISTER
007426  012702  013777                  MOV     #MSG7,R2        ;"RAISE OUTPUT STACKER PRESSURE ARM ABOVE HORIZONTAL THEN LO
WER IT"
007432  005767  171154                  TST     FLAG            ;CHANGE MESSAGE FOR DOCUMATION READER?
007436  001402                          BEQ     .+6             ;NO
007440  012702  014115                  MOV     #MSG7A,R2       ;"LOWER OUTPUT STACKER PLATE TO BOTTOM"
007444  004767  002334                  JSR     %7,TOUT
007450  012702  013467                  MOV     #MSG2,R2        ;"THEN HIT 'CONTINUE' ON THE CONSOLE"
007454  004767  002324                  JSR     %7,TOUT
007460  004767  002442                  JSR     %7,CRLF4        ;MOVE MESSAGE UP ON TTY
007464  000000                          HALT
007466  032713  000400                  BIT     #400,@CRS       ;CHECK BIT 8
007472  001001                          BNE     .+4             ;BRANCH IF SET
007474  104000                          HLT                     ;OFF-LINE (BIT 8) WASN'T SET
007476  005713                          TST     @CRS            ;CHECK SPECIAL CONDITION BIT
007500  100401                          BMI     .+4             ;BRANCH IF SET
007502  104000                          HLT                     ;SPECIAL CONDITION NOT SET
007504  012702  013350                  MOV     #MSG1,R2        ;"PRESS CARD READER 'MOTOR START' AND 'READ START'"
```

```
007510  005767  171076          TST     FLAG            ;CHANGE MESSAGE FOR DOCUMATION READER?
007514  001402                  BEQ     .+6             ;NO
007516  012702  013433          MOV     #MSG1A,R2       ;"PRESS CARD READER 'RESET'"
007522  004767  002256          JSR     %7,TOUT
007526  012702  013467          MOV     #MSG2,R2
007532  004767  002246          JSR     %7,TOUT         ;"THEN HIT 'CONTINUE' ON THE CONSOLE"
007536  004767  002364          JSR     %7,CRLF4        ;MOVE MESSAGE UP ON TTY
007542  000000                  HALT
007544  032713  000400          BIT     #400,@CRS       ;WAIT FOR OFF-LINE TO CLEAR
007550  001375                  BNE     .-4

                        ;A FEED ERROR SHOULD SET BIT 15
                        ;THIS ERROR OCCURS WHEN THE FEED MECHANISM FAILS TO DELIVER A CARD TO THE READ STATION
007552  104400          TESTD1: SCOPE
007554  004767  001514          JSR     %7,INIT
007560  012702  013662          MOV     #MSG5,R2        ;"REMOVE ALL CARDS FROM THE INPUT HOPPER"
007564  004767  002214          JSR     %7,TOUT
007570  012702  013467          MOV     #MSG2,R2        ;"THEN HIT 'CONTINUE' ON THE CONSOLE"
007574  004767  002204          JSR     %7,TOUT
007600  012702  014164          MOV     #MSG8,R2        ;"HOLD DOWN THE SWITCH AT THE BOTTOM OF INPUT HOPPER
007604  005767  171002          TST     FLAG            ;CHANGE MESSAGE FOR DOCUMATION READER?
007610  001402                  BEQ     .+6             ;NO
007612  012702  014255          MOV     #MSG8A,R2       ;"LIFT SWITCH UNDER RIFFLE CAP
007616  004767  002162          JSR     %7,TOUT
007622  012702  013350          MOV     #MSG1,R2        ;"PRESS CARD READER 'MOTOR START' AND 'READ START'
007626  005767  170760          TST     FLAG            ;CHANGE MESSAGE FOR DOCUMATION READER?
007632  001402                  BEQ     .+6             ;NO
007634  012702  013433          MOV     #MSG1A,R2       ;"PRESS CARD READER 'RESET'"
007640  004767  002140          JSR     %7,TOUT
007644  004767  002256          JSR     %7,CRLF4        ;MOVE MESSAGE UP ON TTY
007650  000000                  HALT
007652  032713  002000          BIT     #2000,@CRS      ;WAIT FOR CARD READER TO COME ON-LINE
007656  001775                  BEQ     .-4
007660  004767  001410          JSR     %7,INIT         ;INITIALIZE STATUS REGISTER
007664  012713  000003          MOV     #3,@CRS         ;SET EJECT AND READ
007670  005227  000000          INC     #0              ;WAIT AWHILE
007674  001375                  BNE     .-4
007676  005227  000000          INC     #0
007702  001375                  BNE     .-4
007704  005227  000000          INC     #0
007710  001375                  BNE     .-4
007712  005227  000000          INC     #0
007716  001375                  BNE     .-4
007720  032713  000400          BIT     #400,@CRS       ;TEST OFF-LINE BIT
007724  001001                  BNE     .+4             ;BRANCH IF SET
007726  104000                  HLT                     ;BIT 8 WAS NOT SET
007730  005713                  TST     @CRS            ;CHECK BIT 15
007732  100401                  BMI     .+4             ;BRANCH IF SET
007734  104000                  HLT                     ;BIT 15 WAS NOT SET
007736  012702  013733          MOV     #MSG6,R2
007742  004767  002036          JSR     %7,TOUT         ;"RESTORE CARDS IN THE INPUT HOPPER"
007746  012702  013350          MOV     #MSG1,R2        ;"PRESS CARD READER 'MOTOR START' AND 'READ START'"
007752  005767  170634          TST     FLAG            ;CHANGE MESSAGE FOR DOCUMATION READER?
007756  001402                  BEQ     .+6             ;NO
007760  012702  013433          MOV     #MSG1A,R2       ;"PRESS CARD READER 'RESET'"
```

```
007764  004767  002014                  JSR     %7,TOUT
007770  012702  013467                  MOV     #MSG2,R2        ;"THEN HIT 'CONTINUE' ON THE CONSOLE"
007774  004767  002004                  JSR     %7,TOUT
010000  004767  002122                  JSR     %7,CRLF4        ;MOVE MESSAGE UP ON TTY
010004  000000                          HALT
010006  032713  000400                  BIT     #400,@CRS       ;WAIT FOR OFF-LINE TO CLEAR
010012  001375                          BNE     .-4
010014  005767  170572                  TST     FLAG            ;SKIP NEXT TEST IF DOCUMATION READER
010020  001402                          BEQ     .+6
010022  000167  000314                  JMP     TESTG

                                ;A MOTION ERROR SHOULD SET BIT 15
                                ;THIS ERROR OCCURS WHEN A CARD JAM OCCURS AT THE READ STATION
010026  104400                  TESTE:  SCOPE
010030  004767  001240                  JSR     %7,INIT         ;INITIALIZE STATUS REGISTER
010034  012702  013534                  MOV     #MSG3,R2        ;"PRESS CARD READER 'READ STOP'"
010040  004767  001740                  JSR     %7,TOUT
010044  012702  013467                  MOV     #MSG2,R2        ;"THEN HIT 'CONTINUE' ON THE CONSOLE"
010050  004767  001730                  JSR     %7,TOUT
010054  012702  014314                  MOV     #MSG9,R2        ;"BLOCK THE CARD READER STATION TO
010060  004767  001720                  JSR     %7,TOUT         ;PREVENT A CARD GOING THRU, AND"
010064  012702  013350                  MOV     #MSG1,R2        ;"PRESS CARD READER 'MOTOR START' AND 'READ START'"
010070  004767  001710                  JSR     %7,TOUT
010074  004767  002026                  JSR     %7,CRLF4        ;MOVE MESSAGE UP ON TTY
010100  000000                          HALT
010102  032713  002000                  BIT     #2000,@CRS      ;MONITOR ON-LINE TRANSITION (BIT 10)
010106  001775                          BEQ     .-4             ;CONTINUE WHEN CARD READER COMES ON-LINE
010110  012713  000003                  MOV     #3,@CRS         ;READ A CARD AND SET EJECT
010114  032713  140000                  BIT     #140000,@CRS    ;CHECK DONE AND SPECIAL CONDITION BITS
010120  001775                          BEQ     .-4             ;WAIT
010122  005713                          TST     @CRS            ;CHECK SPECIAL CONDITION BIT
010124  100401                          BMI     .+4             ;CONTINUE IF SET
010126  104000                          HLT                     ;SPECIAL CONDITION NOT SET
010130  012702  014416                  MOV     #MSG10,R2       ;"REMOVE JAMMED CARD"
010134  004767  001644                  JSR     %7,TOUT
010140  012702  013350                  MOV     #MSG1,R2        ;"PRESS CARD READER 'MOTOR START' AND 'READ START'"
010144  004767  001634                  JSR     %7,TOUT
010150  012702  013467                  MOV     #MSG2,R2        ;"THEN HIT 'CONTINUE' ON THE CONSOLE"
010154  004767  001624                  JSR     %7,TOUT
010160  004767  001742                  JSR     %7,CRLF4        ;MOVE MESSAGE UP ON TTY
010164  000000                          HALT
010166  032713  000400                  BIT     #400,@CRS       ;WAIT FOR OFF-LINE TO CLEAR
010172  001375                          BNE     .-4

                                ;A STACK FAIL ERROR SHOULD SET BIT 15
                                ;ERROR OCCURS WHEN 3 CARDS IN A ROW HAVE NOT BEEN DELIVERED PROPERLY TO THE OUTPUT STACK
010174  104400                  TESTF:  SCOPE
010176  004767  001072                  JSR     %7,INIT         ;INITIALIZE STATUS REGISTER
010202  012702  013534                  MOV     #MSG3,R2        ;"PRESS CARD READER 'READ STOP'"
010206  004767  001572                  JSR     %7,TOUT
010212  012702  013467                  MOV     #MSG2,R2        ;"THEN HIT 'CONTINUE' ON THE CONSOLE"
010216  004767  001562                  JSR     %7,TOUT
010222  012702  014443                  MOV     #MSG11,R2       ;"HOLD THE OUTPUT STACKER GATE OPEN.  THEN"
010226  004767  001552                  JSR     %7,TOUT
010232  012702  013350                  MOV     #MSG1,R2        ;"PRESS CARD READER 'MOTOR START' AND
```

```
010236  004767  001542                  JSR     %7,TOUT         ;'READ START,'"
010242  004767  001660                  JSR     %7,CRLF4        ;MOVE MESSAGE UP ON TTY
010246  000000                          HALT
010250  032713  002000                  BIT     #2000,@CRS      ;WAIT FOR CARD READER TO COME ON-LINE
010254  001775                          BEQ     .-4
010256  012701  000003                  MOV     #3,COUNT        ;INITIALIZE COUNTER TO READ 3 CARDS
010262  012713  000003          LOOPF:  MOV     #3,@CRS         ;EJECT A CARD
010266  032713  140000                  BIT     #140000,@CRS    ;WAIT FOR CARD DONE OR SPECIAL CONDITION
010272  001775                          BEQ     .-4
010274  005301                          DEC     COUNT           ;COUNT DOWN
010276  001371                          BNE     LOOPF           ;READ 3 CARDS ALL TOGETHER
010300  005713                          TST     @CRS            ;CHECK SPECIAL CONDITION BIT 15
010302  100401                          BMI     .+4             ;BRANCH IF SET
010304  104000                          HLT                     ;SPECIAL CONDITION NOT SET
010306  012702  013350                  MOV     #MSG1,R2        ;"PRESS CARD READER 'MOTOR START' AND 'READ START'"
010312  004767  001466                  JSR     %7,TOUT
010316  012702  013467                  MOV     #MSG2,R2        ;"THEN HIT 'CONTINUE' ON THE CONSOLE"
010322  004767  001456                  JSR     %7,TOUT
010326  004767  001574                  JSR     %7,CRLF4        ;MOVE MESSAGE UP ON TTY
010332  000000                          HALT
010334  032713  000400                  BIT     #400,@CRS       ;WAIT FOR OFF-LINE TO CLEAR
010340  001375                          BNE     .-4

                                ;DARK-LIGHT ERROR SHOULD SET BIT 15
                                ;THIS OCCURS WHEN DATA IS SENSED BEFORE COLUMN ONE OR AFTER COLUMN EIGHTY
                                ;OR WHEN THE SENSORS ARE NOT ALL SENSING A HOLE AFTER THE CARD HAS PASSED
                                ;THIS TEST IS SKIPPED IF BIT 0 OF THE SWITCH REGISTER EQUALS ONE
                                ;TO MAKE THE 2 DARK-LIGHT CHECK CARDS;
                                ;       1,  TEAR A SMALL PIECE FROM THE LEADING EDGE OF ONE CARD
                                ;       2,  TAPE 2 CARDS TOGETHER TO MAKE ONE "LONG" CARD-IT ONLY NEEDS TO BE
                                ;           ABOUT 1/2 INCH LONGER THAN A REGULAR CARD
010342  104400                  TESTG:  SCOPE
010344  032767  000001  167216          BIT     #1,SR           ;CHECK SW0
010352  001410                          BEQ     CONTG           ;RUN TEST IF NOT SET
010354  004767  000742                  JSR     %7,BELL         ;IF SET, RING BELL AND
010360  000000                          HALT                    ;HALT
010362  012767  007134  001412          MOV     #TESTA+2,RETURN ;SETUP SCOPE LOOP RETURN ADDRESS TO LOOP THRU TESTS
010370  000167  176536                  JMP     TESTA           ;START ERROR TESTS OVER ON CONTINUING
010374  004767  000674          CONTG:  JSR     %7,INIT         ;INITIALIZE STATUS REGISTER
010400  005001                          CLR     COUNT           ;INITIALIZE COUNTER
010402  005201                          INC     COUNT           ;SET TO INDICATE FIRST PASS
010404  012702  014515                  MOV     #MSG12,R2       ;"PLACE SPECIAL DARK-LIGHT CHECK CARDS (SEE LISTING, TESTG)
010410  004767  001370                  JSR     %7,TOUT         ;AT THE BOTTOM OF THE INPUT STACK"
010414  012702  013350          LOOPG:  MOV     #MSG1,R2        ;"PRESS CARD READER 'MOTOR START' AND 'READ START'"
010420  005767  170166                  TST     FLAG            ;CHANGE MESSAGE FOR DOCUMATION READER?
010424  001402                          BEQ     .+6             ;NO
010426  012702  013433                  MOV     #MSG1A,R2       ;"PRESS CARD READER 'RESET'"
010432  004767  001346                  JSR     %7,TOUT
010436  012702  013467                  MOV     #MSG2,R2
010442  004767  001336                  JSR     %7,TOUT         ;"THEN HIT 'CONTINUE' ON THE CONSOLE"
010446  004767  001454                  JSR     %7,CRLF4        ;MOVE MESSAGE UP ON TTY
010452  000000                          HALT
010454  032713  000400                  BIT     #400,@CRS       ;WAIT FOR OFF-LINE TO CLEAR
010460  001375                          BNE     .-4
010462  012713  000003                  MOV     #3,@CRS         ;EJECT THE CARD
```

```
010466  032713  140000          BIT     #140000,@CRS    ;WAIT FOR ERROR OR CARD DONE
010472  001775                  BEQ     .-4
010474  005713                  TST     @CRS            ;CHECK SPECIAL CONDITION
010476  100401                  BMI     .+4             ;CONTINUE IF SET
010500  104000                  HLT                     ;SPECIAL CONDITION NOT SET
010502  005301                  DEC     COUNT           ;COUNT DOWN
010504  001743                  BEQ     LOOPG           ;IF FIRST PASS, LOOP
010506  004767  000610          JSR     %7,BELL         ;RING BELL
010512  000000                  HALT
010514  012702  013350          MOV     #MSG1,R2        ;"PRESS CARD READER 'MOTOR START' AND 'READ START'"
010520  005767  170066          TST     FLAG            ;CHANGE MESSAGE FOR DOCUMATION READER?
010524  001402                  BEQ     .+6             ;NO
010526  012702  013433          MOV     #MSG1A,R2       ;"PRESS CARD READER 'RESET' "
010532  004767  001246          JSR     %7,TOUT
010536  012702  013467          MOV     #MSG2,R2        ;"THEN HIT 'CONTINUE' ON THE CONSOLE"
010542  004767  001236          JSR     %7,TOUT
010546  004767  001354          JSR     %7,CRLF4        ;MOVE MESSAGE UP ON TTY
010552  000000                  HALT
010554  032713  000400          BIT     #400,@CRS       ;WAIT FOR OFF-LINE TO CLEAR
010560  001375                  BNE     .-4
010562  012767  007134  001212  MOV     #TESTA+2,RETURN ;SETUP SCOPE LOOP RETURN ADDRESS
010570  000167  176336          JMP     TESTA           ;LOOP THRU TEST ON CONTINUING


                                ;ROUTINE TO LOOP THRU A SINGLE INSTRUCTION TEST
                                ;NOTE THAT SW11 MUST BE DOWN AFTER 2ND HALT
010574  004767  170032  TESTX:  JSR     %7,SETUP                ;SETUP POINTERS AND FLAGS
010600  000000                  HALT                            ;WAIT FOR STARTING ADDRESS
010602  016767  166762  000064  MOV     SR,RETRNX               ;STORE STARTING ADDRESS
010610  062767  000002  000056  ADD     #2,RETRNX               ;CHANGE TO FIRST ADDRESS AFTER SCOPE INSTRUCTION
010616  000000                  HALT                            ;SET SR OPTIONS (BIT 11 MUST = 0)
010620  032767  010000  166742  BIT     #10000,SR               ;CHECK SW12
010626  001404                  BEQ     .+12                    ;BRANCH IF NOT SET
010630  042767  000020  167140  BIC     #20,PSR                 ;CLEAR TRACE BIT
010636  000403                  BR      .+10                    ;SKIP NEXT INSTRUCTION
010640  052767  000020  167130  BIS     #20,PSR                 ;SET TRACE BIT
010646  005067  001126          CLR     ITCNT                   ;CLEAR ITERATION COUNTER
010652  012767  010664  001122  MOV     #XLOOP,RETURN           ;LOAD RETURN ADDRESS
010660  000177  000010          JMP     @RETRNX                 ;JUMP TO TEST
010664  005067  001110  XLOOP:  CLR     ITCNT                   ;KEEP ITERATION COUNTER AT ZERO
010670  000177  000000          JMP     @RETRNX                 ;JUMP TO TEST
010674  000000          RETRNX: 0


                                ;ROUTINE TO CHECK CARDS WHICH HAVE ALL COLUMNS IDENTICALLY PUNCHED.
                                ;THIS ROUTINE ALLOWS SPECIFIC TYPES OF DATA FAILURES TO BE STUDIED
                                ;EASILY.  THE ROUTINE HALTS ONCE AT THE START.  SET THE CORRECT CARD
                                ;IMAGE PATTERN IN SW11-SW00, THEN HIT CONTINUE (AFTER THE DECK IS
                                ;LOADED AND CARD READER IS ON-LINE).  THE PATTERN IS STORED, AND THEN
                                ;EACH COLUMN OF EACH CARD IS READ TWICE AND COMPARED WITH IT.  IF A
                                ;DISCREPANCY OCCURS, THE ERROR IS PRINTED OUT ALONG WITH THE TOTAL
                                ;NUMBER OF CARDS READ AND THE TOTAL NUMBER OF DATA ERRORS DISCOVERED
                                ;UP TO THAT POINT (ALL PRINTOUTS ARE IN OCTAL).  WHEN THE INPUT HOPPER
                                ;IS EMPTY, THE ROUTINE RINGS THE BELL AND WAITS FOR MORE CARDS TO BE
                                ;LOADED AND THE CARD READER TO BE PUT BACK ON-LINE.
```

```
                                        ;SW15=1 CAUSES A HALT AFTER AN ERROR, AND SW13=1 INHIBITS ERROR PRINTOUTS.

010676  004767  167730          CKSAME: JSR     %7,SETUP        ;INITIALIZE POINTERS
010702  000000                          HALT                    ;WAIT FOR CARD IMAGE PATTERN
010704  016767  166660  000360          MOV     SR,CARDIM       ;STORE PATTERN
010712  042767  170000  000352          BIC     #170000,CARDIM  ;CLEAR UPPER BITS OF PATTERN
010720  005067  000344                  CLR     TOTCRD          ;INITIALIZE CARD COUNT
010724  005067  000336                  CLR     TOTERR          ;INITIALIZE ERROR COUNT
010730  005067  167674                  CLR     ERFLG           ;CLEAR FLAG FOR PRINTING ERROR HEADING
010734  005067  175614          CKLOOP: CLR     CLCNT           ;INITIALIZE COLUMN COUNT
010740  032713  000400                  BIT     #400,@CRS       ;CHECK BIT 8
010744  001017                          BNE     CKSIT           ;BRANCH IF SET TO WAIT FOR READER TO COME ON-LINE.
010746  005213                          INC     @CRS            ;START READING CARD
010750  005267  000314                  INC     TOTCRD          ;INCREMENT CARD COUNT
010754  105713          CKLP1:  TSTB    @CRS            ;CHECK COLUMM READY
010756  100426                          BMI     CKCOL           ;BRANCH IF SET
010760  032713  040000                  BIT     #40000,@CRS     ;CHECK CARD DONE
010764  001015                          BNE     CKCRD           ;BRANCH IF SET
010766  005713                          TST     @CRS            ;CHECK SPECIAL CONDITION
010770  100371                          BPL     CKLP1           ;LOOP IF NOT SET
010772  032713  000400                  BIT     #400,@CRS       ;CHECK BIT 8
010776  001002                          BNE     CKSIT           ;BRANCH IF SET TO WAIT FOR READER ON-LINE.
011000  104000                          HLT                     ;SPECIAL CONDITION SET, BIT 8 CLEAR
011002  000754                          BR      CKLOOP

011004  004767  000312          CKSIT:  JSR     %7,BELL         ;RING BELL TO SIGNIFY READER OFF-LINE
011010  032713  000400          CKSIT1: BIT     #400,@CRS       ;CHECK BIT 8
011014  001375                          BNE     CKSIT1          ;LOOP IF STILL SET
011016  000746                          BR      CKLOOP          ;START NEXT CARD
011020  022767  000120  175526  CKCRD:  CMP     #80.,,CLCNT     ;CHECK FOR 80 COLUMNS READ
011026  001742                          BEQ     CKLOOP          ;START NEXT CARD IF OK
011030  104000                          HLT                     ;FINAL COLUMN COUNT WASN'T 80
011032  000740                          BR      CKLOOP          ;START NEXT CARD
011034  011467  175516          CKCOL:  MOV     @CRB1,DAT1      ;READ DATA BUFFER
011040  005267  175510                  INC     CLCNT           ;COUNT COLUMNS
011044  105713                          TSTB    @CRS            ;CHECK COLUMN READY
011046  100002                          BPL     .+6             ;BRANCH IF OK
011050  104000                          HLT                     ;READING DBR DIDN'T CLEAR READY
011052  000730                          BR      CKLOOP          ;START NEXT CARD AFTER ERROR
011054  012701  000200                  MOV     #200,COUNT      ;WAIT AWHILE

011060  005301          CKLP2:  DEC     COUNT
011062  001376                  BNE     CKLP2
011064  011467  175470          MOV     @CRB1,DAT2      ;READ CRB1 AGAIN
011070  026767  175462  000174  CMP     DAT1,CARDIM     ;COMPARE FIRST DATA TO PATTERN
011076  001005                  BNE     CKFAIL          ;BRANCH IF FAILURE
011100  026767  175454  000164  CMP     DAT2,CARDIM     ;COMPARE SECOND READING TO PATTERN
011106  001001                  BNE     CKFAIL          ;BRANCH IF FAILURE
011110  000721                  BR      CKLP1           ;WAIT FOR NEXT COLUMN OR END OF CARD
011112  005267  000150  CKFAIL: INC     TOTERR          ;COUNT ERRORS
011116  032767  020000  166444  BIT     #20000,SR       ;CHECK FOR INHIBITING PRINTOUT
011124  001047                  BNE     CKHLT           ;BRANCH AROUND PRINTOUT IF SET
011126  005767  167476          TST     ERFLG           ;TEST FLAG TO PRINT HEADING
011132  001006                  BNE     CKNOHD          ;BRANCH IF ALREADY DONE
011134  005267  167470          INC     ERFLG           ;PRINT HEADING ONCE ONLY
```

```
011140  012702  015110          MOV     #MSG19,R2       ;OUTPUT HEADING
011144  004767  000634          JSR     %7,TOUT
011150  004767  000720  CKNOHD: JSR     %7,CRLF         ;OUTPUT CARRIAGE RETURN, LINEFEED
011154  016702  175374          MOV     CLCNT,R2        ;PRINT COLUMN NUMBER
011160  004767  000404          JSR     %7,PROCT
011164  004767  000212          JSR     %7,SPACE
011170  016702  175362          MOV     DAT1,R2         ;PRINT FIRST READING
011174  004767  000370          JSR     %7,PROCT
011200  004767  000176          JSR     %7,SPACE
011204  016702  175350          MOV     DAT2,R2         ;PRINT SECOND READING
011210  004767  000354          JSR     %7,PROCT
011214  004767  000162          JSR     %7,SPACE
011220  016702  000044          MOV     TOTCRD,R2       ;PRINT TOTAL NUMBER OF CARDS READ
011224  004767  000340          JSR     %7,PROCT
011230  004767  000146          JSR     %7,SPACE
011234  016702  000026          MOV     TOTERR,R2       ;PRINT TOTAL NUMBER OF DATA ERRORS
011240  004767  000324          JSR     %7,PROCT
011244  005767  166320  CKHLT: TST     SR              ;CHECK SW15 TO HALT ON ERROR
011250  100002                  BPL     CKDONE          ;BRANCH IF NOT SET
011252  000000                  HALT                    ;HALT ON ERROR
011254  000627                  BR      CKLOOP          ;CONTINUE
011256  032713  140000  CKDONE: BIT     #140000,@CRS    ;WAIT FOR SPECIAL CONDITION OR DONE
011262  001775                  BEQ     CKDONE
011264  000623                  BR      CKLOOP          ;START NEXT CARD AFTER CHECKING BIT 8
011266  000000          TOTERR: 0
011270  000000          TOTCRD: 0
011272  000000          CARDIM: 0


                        ;ISSUE MESSAGE IF CARD READER IS OFF-LINE
                        ;WAIT FOR BUSY TO CLEAR IN CASE CARD READER IS STILL READING A CARD
                        ;INITIALIZE STATUS REGISTER AND USE ERROR HALT IF IT DOESN'T CLEAR PROPERLY
                        ;NOTE THAT PROGRAM WILL HANG HERE IF BUSY REMAINS SET
011274  004767  000046  INIT:  JSR     %7,CKBIT8       ;SEE IF OFF-LINE BIT IS SET
011300  032713  001000          BIT     #1000,@CRS      ;WAIT FOR BUSY TO CLEAR, IN CASE
011304  001375                  BNE     .-4             ;A CARD IS STILL BEING READ
011306  005013                  CLR     @CRS            ;INITIALIZE STATUS REGISTER
011310  005714                  TST     @CRB1           ;READ DATA BUFFER TO CLEAR COLUMN READY
011312  005713                  TST     @CRS            ;MAKE SURE INITIALIZATION OK
011314  001401                  BEQ     .+4             ;BRANCH IF ALL BITS ZERO
011316  104000                  HLT                     ;NOT ALL BITS OF STATUS REGISTER ARE ZERO
011320  000207                  RTS     %7              ;RETURN

                        ;BELL ON PASS COMPLETE
011322  105777  167260  BELL:  TSTB    @TCSR           ;WAIT FOR TTY READY
011326  100375                  BPL     .-4
011330  012777  000207  167252  MOV     #207,@TDBR      ;RING BELL
011336  012767  000001  000432  MOV     #1,ITMAX        ;MAKE CERTAIN ITERATION MAXIMUM IS CORRECT
011344  000207                  RTS     %7              ;RETURN

                        ;SUBROUTINE TO CHECK FOR BIT 8 (OFF-LINE) BEING SET IN CARD
                        ;READER CSR, AND PRINT OUT A MESSAGE IF IT IS
```

```
011346  032713  000400          CKBIT8: BIT     #400,@CRS       ;CHECK BIT 8
011352  001001                          BNE     .+4             ;BRANCH IF SET
011354  000207                          RTS     %7              ;RETURN IF NOT SET
011356  012702  015070                  MOV     #MSG18,R2       ;OUTPUT MESSAGE
011362  004767  000416                  JSR     %7,TOUT         ;"BIT 8 WAS SET"
011366  012702  015005                  MOV     #MSG17,R2       ;"REMEDY THE ERROR CONDITION
011372  004767  000406                  JSR     %7,TOUT         ;AND PRESS 'CONTINUE'"
011376  000000                          HALT                    ;WAIT FOR CONTINUE
011400  000762                          BR      CKBIT8          ;CHECK AGAIN

                                ;SUBROUTINE TO ISSUE N SPACES
                                ;N IS ONE PLUS VALUE CONTAINED IN SPACEX
                                ;SPACEX IS CLEARED WITHIN THE SUBROUTINE, SO THAT A CALL ON
                                ;SPACE WITHOUT LOADING SPACEX ISSUES ONLY ONE SPACE
011402  105777  167200          SPACE:  TSTB    @TCSR           ;WAIT FOR TTY READY
011406  100375                          BPL     .-4
011410  012777  000240  167172          MOV     #240,@TDBR      ;OUTPUT A SPACE
011416  005367  000010                  DEC     SPACEX          ;DECREMENT COUNT
011422  100367                          BPL     SPACE           ;LOOP IF NOT DONE
011424  005067  000002                  CLR     SPACEX          ;RESET COUNT TO ZERO
011430  000207                          RTS     %7              ;RETURN
011432  000000          SPACEX: 0



                                ;ENTERED WITH SYSTEM TRAP CALL (HLT)
                                ;PRINT OUT THE ERROR PC AND STATUS REGISTER
011434  036727  166130  020000  PRINT:  BIT     SR,     #20000  ;TEST FOR INHIBIT PRINT OUT
011442  001401                          BEQ     .+4             ;BRANCH TO PRINT
011444  000437                          BR      B,CK            ;INHIBIT, CHECK FOR HALT
011446  012667  000112                  MOV     (6)+,   SAVPC   ;PC OF FAILING ROUTINE
011452  012667  000110                  MOV     (6)+,   SAVPSR  ;PSR OR ERROR CONDITION
011456  024646                          CMP     -(6),   -(6)    ;RESTORE STACK
011460  004767  000410                  JSR     %7,CRLF         ;OUTPUT CARRIAGE RETURN, LINEFEED
011464  010267  000066                  MOV     %2,     SAVR2   ;SAVE R2
011470  016702  000070                  MOV     SAVPC,  %2
011474  004767  000070                  JSR     %7,     PROCT   ;PRINT PC+2 IN OCTAL
011500  105777  167102                  TSTB    @TCSR           ;WAIT FOR TTY READY
011504  100375                          BPL     .-4
011506  012777  000240  167074          MOV     #240,   @TDBR   ;OUTPUT A SPACE
011514  016702  000046                  MOV     SAVPSR, %2
011520  004767  000044                  JSR     %7,     PROCT   ;PRINT PROCESSOR STATUS AT TIME OF FAILURE
011524  016702  000026                  MOV     SAVR2,  %2      ;RESTORE REGISTER 2
011530  105777  167052                  TSTB    @TCSR           ;WAIT FOR TTY READY
011534  100375                          BPL     .-4
011536  012777  000240  167044          MOV     #240,@TDBR
011544  005767  166020          B,CK:   TST     SR              ;CHECK SR FOR HALT SWITCH
011550  100001                          BPL     .+4             ;BRANCH IF NOT SET
011552  000000                          HALT                    ;HALT ON ERROR UP
011554  000002                          RTI                     ;RETURN TO MAIN LINE
011556  000000          SAVR2:  0
011560  000000          SAVR3:  0
011562  000000          SAVR4:  0
011564  000000          SAVPC:  0
```

```
011566  000000                  SAVPSR: 0

011570  010367  177764          PROCT:  MOV     %3,SAVR3        ;SAVE R3
011574  010467  177762                  MOV     %4,SAVR4        ;SAVE R4
011600  005004                          CLR     %4              ;CLEAR R4 TO USE AS COUNTER
011602  005001                          CLR     COUNT           ;CLEAR COUNT TO USE AS CARRY FLAG
011604  012703  000260                  MOV     #260,   %3      ;SETUP ASCII ZERO IN R3
011610  005702                          TST     %2              ;CHECK BIT 15 OF DESIRED NUMBER
011612  100001                          BPL     .+4             ;BRANCH IF NOT SET
011614  005203                          INC     %3              ;CHANGE TO ASCII ONE
011616  006102                          ROL     %2              ;ROTATE INTO RIGHTMOST BIT
011620  006102                          ROL     %2              ;TO PREPARE FOR LOOP
011622  005501                          ADC     COUNT           ;STORE CARRY
011624  105777  166756          C,WAIT: TSTB    @TCSR           ;WAIT FOR TTY READY
011630  100375                          BPL     C,WAIT
011632  010377  166752                  MOV     %3,     @TDBR   ;OUTPUT ASCII
011636  005204                          INC     %4              ;COUNT CHARACTERS OUTPUT
011640  020427  000006                  CMP     %4,     #6      ;CHECK FOR DONE
011644  001005                          BNE     C,CONT          ;BRANCH IF NOT DONE
011646  016703  177706                  MOV     SAVR3,%3        ;RESTORE REGISTER 3
011652  016704  177704                  MOV     SAVR4,%4        ;RESTORE REGISTER 4
011656  000207                          RTS     %7              ;RETURN
011660  000241          C,CONT: CLC                             ;CLEAR CARRY
011662  005701                          TST     COUNT           ;TEST CARRY FLAG
011664  001402                          BEQ     .+6             ;BRANCH IF NOT SET
011666  005001                          CLR     COUNT           ;CLEAR FLAG
011670  000261                          SEC                     ;SET CARRY
011672  006102                          ROL     %2              ;ROTATE NEXT 3 BITS INTO RIGHTMOST 3
011674  006102                          ROL     %2
011676  006102                          ROL     %2
011700  005501                          ADC     COUNT           ;STORE CARRY
011702  010203                          MOV     %2,     %3      ;MOVE DATA FOR OUTPUT
011704  042703  177770                  BIC     #177770,%3      ;CLEAR ALL BUT RIGHTMOST 3 BITS
011710  052703  000260                  BIS     #260,   %3      ;SET TO ASCII EQUIVALENT
011714  000743                          BR      C,WAIT          ;LOOP

                                ;SCOPE AND/OR ITERATION LOOP FOR EACH TEST 2 TIMES
011716  032767  040000  165644  SCOPEC: BIT     #40000, SR      ;TEST SR FOR SCOPE
011724  001012                          BNE     D,1             ;YES,SCOPE
011726  032767  004000  165634          BIT     #4000,SR        ;NO- TEST FOR ITERATION
011734  001013                          BNE     D,2             ;INHIBIT ITERATION
011736  026767  000036  000032          CMP     ITCNT,ITMAX     ;CHECK FOR ITERATIONS COMPLETE
011744  100007                          BPL     D,2             ;EXIT-DONE
011746  005267  000026                  INC     ITCNT           ;INCREMENT COUNT
011752  022626          D,1:    CMP     (6)+,   %6      ;REPOSITION STACK POINTER
011754  012667  166016                  MOV     (6)+,   PSR     ;RESTORE PROCESSOR STATUS
011760  000177  000016                  JMP     @RETURN         ;RETURN TO RERUN TEST
011764  005067  000010          D,2:    CLR     ITCNT           ;CLEAR COUNTER
011770  011667  000006                  MOV     @%6,    RETURN  ;SAVE SCOPE RETURN POINTER
011774  000002                          RTI                     ;RETURN INLINE-NEXT TEST
011776  000001          ITMAX:  1                               ;MAX NUMBER OF ITERATIONS
012000  000000          ITCNT:  0                               ;COUNT LOCATION FOR ITERATION LOOP
012002  000746          RETURN: TEST1+2                         ;ADDRESS OF LAST TEST

                                ;MOV ADDRESS OF MESSAGE TO REGISTER 2
```

```
                        ;THEN JSR %7, TOUT
012004  142777  000177  166574  TOUT:   BICB    #177,   @TCSR   ;CLEAR INT FLAG
012012  111267  000054          MOVB    @%2,    L,EOMK  ;MOVE IN EOM MARKER
012016  005202                  L,INC:  INC     %2              ;MOVE DATA POINTER TO NEXT BYTE
012020  121267  000046          L,TOUT: CMPB    @%2,    L,EOMK  ;COMPARE FOR EOM
012024  001006                          BNE     L,CNT           ;BRANCH IF NOT END OF MESSAGE
012026  105777  166554                  TSTB    @TCSR           ;WAIT FOR TTY READY
012032  100375                          BPL     .-4
012034  005077  166550                  CLR     @TDBR           ;OUTPUT NULL
012040  000207                          RTS     %7              ;RETURN IF EOM
012042  121227  000100          L,CNT:  CMPB    @%2,    #'@     ;CHECK FOR CR,LF REQUEST
012046  001003                          BNE     .+10            ;BRANCH IF NOT
012050  004767  000020                  JSR     %7,CRLF         ;OUTPUT CARRIAGE RETURN, LINEFEED
012054  000760                          BR      L,INC           ;LOOP
012056  105777  166524                  TSTB    @TCSR           ;WAIT FOR TTY
012062  100375                          BPL     .-4
012064  112277  166520                  MOVB    (2)+,   @TDBR   ;OUTPUT NEXT CHARACTER
012070  000753                          BR      L,TOUT          ;CONTINUE
012072  000000          L,EOMK: 0

                        ;SUBROUTINE TO ISSUE CARRIAGE RETURN AND LINEFEED
012074  105777  166506  CRLF:   TSTB    @TCSR           ;WAIT FOR TTY READY
012100  100375                  BPL     .-4
012102  112777  000215  166500          MOVB    #215,   @TDBR   ;SEND CARRIAGE RETURN
012110  105777  166472                  TSTB    @TCSR           ;WAIT FOR TTY
012114  100375                          BPL     .-4
012116  112777  000212  166464          MOVB    #212,   @TDBR   ;SEND LINE FEED
012124  000207                          RTS     %7              ;RETURN


                        ;DO 4 CRLF'S TO MOVE MESSAGES ON TELETYPE
012126  004767  177742  CRLF4:  JSR     %7,CRLF
012132  004767  177736          JSR     %7,CRLF
012136  004767  177732          JSR     %7,CRLF
012142  004767  177726          JSR     %7,CRLF
012146  000207                  RTS     %7

                        ;DATA TABLES FOR DATA RELIABILITY TESTS

                        ;ALPHANUMERIC DECK DATA TABLE
                        ;FIRST VALUE FOR A COLUMN IS THE DIRECT
                        ;CARD IMAGE FOR THAT COLUMN ON CARD 1
                        ;THE SECOND VALUE IS THE ENCODED FORM OF THAT DATA
                                                ;COLUMN CHAR HOLLERITH
012150  004000  ALPCD:  4000                    ;1      8       12
012152  000200          200
012154  004420          4400                    ;2      A       12 1
012156  000201          201
012160  004200          4200                    ;3      B       12 2
012162  000202          202
012164  004100          4100                    ;4      C       12 3
012166  000203          203
012170  004040          4040                    ;5      D       12 4
012172  000204          204
012174  004020          4020                    ;6      E       12 5
```

```
012176  000205                  205
012200  004010                  4010          ;7    F       12 6
012202  000206                  206
012204  004004                  4004          ;8    G       12 7
012206  000207                  207
012210  004002                  4002          ;9    H       12 8
012212  000210                  210
012214  004001                  4001          ;10   I       12 9
012216  000220                  220
012220  004202                  4202          ;11   CENT    12 8 2
012222  000212                  212
012224  004102                  4102          ;12   .       12 8 3
012226  000213                  213
012230  004042                  4042          ;13   <       12 8 4
012232  000214                  214
012234  004022                  4022          ;14   (       12 8 5
012236  000215                  215
012240  004012                  4012          ;15   +       12 8 6
012242  000216                  216
012244  004006                  4006          ;16   |       12 8 7
012246  000217                  217
012250  002000                  2000          ;17   &       11
012252  000100                  100
012254  002400                  2400          ;18   J       11 1
012256  000101                  101
012260  002200                  2200          ;19   K       11 2
012262  000102                  102
012264  002100                  2100          ;20   L       11 3
012266  000103                  103
012270  002040                  2040          ;21   M       11 4
012272  000104                  104
012274  002020                  2020          ;22   N       11 5
012276  000105                  105
012300  002010                  2010          ;23   O       11 6
012302  000106                  106
012304  002004                  2004          ;24   P       11 7
012306  000107                  107
012310  002002                  2002          ;25   Q       11 8
012312  000110                  110

012314  002001                  2001          ;26   R       11 9
012316  000120                  120
012320  002202                  2202          ;27   !       11 8 2
012322  000112                  112
012324  002102                  2102          ;28   $       11 8 3
012326  000113                  113
012330  002042                  2042          ;29   *       11 8 4
012332  000114                  114
012334  002022                  2022          ;30   )       11 8 5
012336  000115                  115
012340  002012                  2012          ;31   ;       11 8 6
012342  000116                  116
012344  002006                  2006          ;32   BLANK   11 8 7
012346  000117                  117
012350  001000                  1000          ;33   0       0
```

```
012352  000040                  40
012354  001400                  1400          ;34   /      0 1
012356  000041                  41
012360  001200                  1200          ;35   S      0 2
012362  000042                  42
012364  001100                  1100          ;36   T      0 3
012366  000043                  43
012370  001040                  1040          ;37   U      0 4
012372  000044                  44
012374  001020                  1020          ;38   V      0 5
012376  000045                  45
012400  001010                  1010          ;39   W      0 6
012402  000046                  46
012404  001004                  1004          ;40   X      0 7
012406  000047                  47
012410  001002                  1002          ;41   Y      0 8
012412  000050                  50
012414  001001                  1001          ;42   Z      0 9
012416  000060                  60
012420  001202                  1202          ;43          0 8 2
012422  000052                  52
012424  001102                  1102          ;44   ,      0 8 3
012426  000053                  53
012430  001042                  1042          ;45   %      0 8 4
012432  000054                  54
012434  001022                  1022          ;46   -      0 8 5
012436  000055                  55
012440  001012                  1012          ;47   >      0 8 6
012442  000056                  56
012444  001006                  1006          ;48   ?      0 8 7
012446  000057                  57
012450  000000                  0000          ;49          BLANK
012452  000000                  0
012454  000400                  0400          ;50   1      1
012456  000001                  1
012460  000200                  0200          ;51   2      2
012462  000002                  2
012464  000100                  0100          ;52   3      3
012466  000003                  3
012470  000040                  0040          ;53   4      4

012472  000004                  4
012474  000020                  0020          ;54   5      5
012476  000005                  5
012500  000010                  0010          ;55   6      6
012502  000006                  6
012504  000004                  0004          ;56   7      7
012506  000007                  7
012510  000002                  0002          ;57   8      8
012512  000010                  10
012514  000001                  0001          ;58   9      9
012516  000020                  20
012520  000202                  0202          ;59   !      8 2
012522  000012                  12
012524  000102                  0102          ;60   #      8 3
```

```
012526  000013                  13
012530  000042                  0042                    ;61     A       8 4
012532  000014                  14
012534  000022                  0022                    ;62     '       8 5
012536  000015                  15
012540  000012                  0012                    ;63     =       8 6
012542  000016                  16
012544  000026                  0006                    ;64     "       8 7
012546  000017                  17
012550  004000                  4000                    ;65     &       12
012552  000200                  200
012554  004400                  4400                    ;66     A       12 1
012556  000201                  201
012560  004200                  4200                    ;67     B       12 2
012562  000202                  202
012564  004100                  4100                    ;68     C       12 3
012566  000203                  203
012570  004040                  4040                    ;69     D       12 4
012572  000204                  204
012574  004020                  4020                    ;70     E       12 5
012576  000205                  205
012600  004010                  4010                    ;71     F       12 6
012602  000206                  206
012604  004004                  4004                    ;72     G       12 7
012606  000207                  207
012610  004002                  4002                    ;73     H       12 8
012612  000210                  210
012614  004001                  4001                    ;74     I       12 9
012616  000220                  220
012620  004202                  4202                    ;75     CENT    12 8 2
012622  000212                  212
012624  004102                  4102                    ;76     .       12 8 3
012626  000213                  213
012630  004042                  4042                    ;77     <       12 8 4
012632  000214                  214
012634  004022                  4022                    ;78     (       12 8 5
012636  000215                  215
012640  004012                  4012                    ;79     +       12 8 6
012642  000216                  216
012644  004006                  4006                    ;80     !       12 8 7

012646  000217          ALPEND: 217
                        ;BINARY DECK DATA TABLE
                        ;FIRST VALUE FOR A COLUMN IS THE DIRECT CARD IMAGE OF THAT COLUMN ON CARD1
                        ;THE SECOND VALUE IS THE ENCODED VALUE, WHICH ORS THE OCTAL REPRESENTATION OF
                        ;ROWS ONE THRU SEVEN
012650  000000          BINCD:  0                       ;CARD COLUMN 1
012652  000000                  0
012654  000001                  1                       ;2
012656  000020                  20
012660  000002                  2                       ;3
012662  000010                  10
012664  000004                  4                       ;4
012666  000007                  7
012670  000010                  10                      ;5
```

```
012672  000006                          6
012674  000020                          20                      ;6
012676  000005                          5
012700  000040                          40                      ;7
012702  000004                          4
012704  000100                          100                     ;8
012706  000003                          3
012710  000200                          200                     ;9
012712  000002                          2
012714  000400                          400                     ;10
012716  000001                          1
012720  001000                          1000                    ;11
012722  000040                          40
012724  002000                          2000                    ;12
012726  000144                          100
012730  004000                          4000                    ;13
012732  000200                          200
012734  001111                          1111                    ;14
012736  000067                          67
012740  002222                          2222                    ;15
012742  000117                          117
012744  003333                          3333                    ;16
012746  000177                          177
012750  004444                          4444                    ;17
012752  000227                          207
012754  005555                          5555                    ;18
012756  000267                          267
012760  006666                          6666                    ;19
012762  000317                          317
012764  007777                          7777                    ;20
012766  000377                          377
012770  001010                          1010                    ;21
012772  000046                          46
012774  001212                          1212                    ;22
012776  000056                          56
013000  001313                          1313                    ;23
013002  000077                          77
013004  001414                          1414                    ;24
013006  000047                          47
013010  001515                          1515                    ;25
013012  000067                          67
013014  001616                          1616                    ;26
013016  000057                          57
013020  001717                          1717                    ;27

013022  000077                          77
013024  002020                          2020                    ;28
013026  000105                          105
013030  002121                          2121                    ;29
013032  000127                          127
013034  002323                          2323                    ;30
013036  000137                          137
013040  002424                          2424                    ;31
013042  000107                          107
013044  002525                          2525                    ;32
```

```
013046  000127                          127
013050  002626                          2626                    ;33
013052  000117                          117
013054  002727                          2727                    ;34
013056  000137                          137
013060  003030                          3030                    ;35
013062  000147                          147
013064  003131                          3131                    ;36
013066  000167                          167
013070  003232                          3232                    ;37
013072  000157                          157
013074  003434                          3434                    ;38
013076  000147                          147
013100  003535                          3535                    ;39
013102  000167                          167
013104  003636                          3636                    ;40
013106  000157                          157
013110  003737                          3737                    ;41
013112  000177                          177
013114  004040                          4040                    ;42
013116  000204                          204
013120  004141                          4141                    ;43
013122  000227                          227
013124  004242                          4242                    ;44
013126  000216                          216
013130  004343                          4343                    ;45
013132  000237                          237
013134  004545                          4545                    ;46
013136  000227                          227
013140  004646                          4646                    ;47
013142  000217                          217
013144  004747                          4747                    ;48
013146  000237                          237
013150  005050                          5050                    ;49
013152  000246                          246
013154  005151                          5151                    ;50
013156  000267                          267
013160  005252                          5252                    ;51
013162  000256                          256
013164  005353                          5353                    ;52
013166  000277                          277
013170  005454                          5454                    ;53
013172  000247                          247
013174  005656                          5656                    ;54
013176  000257                          257
013200  005757                          5757                    ;55
013202  000277                          277
013204  006060                          6060                    ;56

013206  000305                          305
013210  006161                          6161                    ;57
013212  000327                          327
013214  006262                          6262                    ;58
013216  000317                          317
013220  006363                          6363                    ;59
```

```
013222  000337                          337
013224  006464                          6464                    ;60
013226  000307                          307
013230  006565                          6565                    ;61
013232  000327                          327
013234  006767                          6767                    ;62
013236  000337                          337
013240  007070                          7070                    ;63
013242  000347                          347
013244  007171                          7171                    ;64
013246  000367                          367
013250  007272                          7272                    ;65
013252  000357                          357
013254  007373                          7373                    ;66
013256  000377                          377
013260  007474                          7474                    ;67
013262  000347                          347
013264  007575                          7575                    ;68
013266  000367                          367
013270  007676                          7676                    ;69
013272  000357                          357
013274  000101                          0101                    ;70
013276  000023                          23
013300  000202                          0202                    ;71
013302  000012                          12
013304  000303                          0303                    ;72
013306  000033                          33
013310  000404                          0404                    ;73
013312  000007                          7
013314  000505                          0505                    ;74
013316  000027                          27
013320  000606                          0606                    ;75
013322  000017                          17
013324  000707                          0707                    ;76
013326  000037                          37
013330  003210                          3210                    ;77
013332  000146                          146
013334  000123                          0123                    ;78
013336  000037                          37
013340  007654                          7654                    ;79
013342  000347                          347
013344  004567                          4567                    ;80
013346  000237                  BINEND: 237

013350      057                 MSG1:   .ASCII ;/@PRESS CARD READER 'MOTOR START' AND 'READ START'/;
013351      100
013352      120
013353      122
013354      105
013355      123
013356      123
013357      040
013360      103
013361      101
013362      122
```

```
013363      104
013364      040
013365      122
013366      105
013367      101
013370      104
013371      105
013372      122
013373      040
013374      047
013375      115
013376      117
013377      124
013400      117
013401      122
013402      040
013403      123
013404      124
013405      101
013406      122
013407      124
013410      047
013411      040
013412      101
013413      116
013414      104
013415      040
013416      047
013417      122
013420      105
013421      101
013422      104
013423      040
013424      123
013425      124
013426      101
013427      122
013430      124
013431      047
013432      057

013433      057                 MSG1A:  .ASCII  /?PRESS CARD READER 'RESET'?/
013434      120
013435      120
013436      122
013437      105
013440      123
013441      123
013442      040
013443      103
013444      101
013445      122
013446      104
013447      040
013450      122
```

```
013451      125
013452      101
013453      104
013454      105
013455      122
013456      040
013457      047
013460      122
013461      125
013462      123
013463      105
013464      124
013465      047
013466      057

013467      057                 MSG2:   .ASCII  /?THEN HIT 'CONTINUE' ON THE CONSOLE?/
013470      120
013471      124
013472      110
013473      105
013474      116
013475      040
013476      110
013477      111
013500      124
013501      040
013502      047
013503      103
013504      117
013505      116
013506      124
013507      111
013510      116
013511      125
013512      105
013513      047
013514      040
013515      117
013516      116
013517      040
013520      124
013521      110
013522      105
013523      040
013524      103
013525      117
013526      116
013527      123
013530      117
013531      114
013532      105
013533      057

013534      057                 MSG3:   .ASCII  /?PRESS CARD READER 'READ STOP'?/
013535      100
```

```
013536      120
013537      122
013540      105
013541      123
013542      123
013543      040
013544      103
013545      101
013546      122
013547      104
013550      040
013551      122
013552      105
013553      101
013554      104
013555      105
013556      122
013557      040
013560      047
013561      122
013562      105
013563      101
013564      104
013565      040
013566      123
013567      124
013570      117
013571      120
013572      047
013573      057

013574      057                 MSG3A:  .ASCII  /?PRESS CARD READER 'STOP'?/
013575      100
013576      120
013577      122
013600      105
013601      123
013602      123
013603      040
013604      103
013605      101
013606      122
013607      104
013610      040
013611      122
013612      105
013613      101
013614      124
013615      105
013616      122
013617      040
013620      047
013621      123
013622      124
013623      117
```

```
013624      120
013625      047
013626      057

013627      057                 MSG4:   .ASCII  /?THE INTERRUPT LEVEL WAS /
013630      120
013631      124
013632      110
013633      105
013634      040
013635      111
013636      116
013637      124
013640      105
013641      122
013642      122
013643      125
013644      120
013645      124
013646      040
013647      114
013650      105
013651      126
013652      105
013653      114
013654      040
013655      127
013656      101
013657      123
013660      040
013661      057

013662      057                 MSG5:   .ASCII  /?REMOVE ALL CARDS FROM THE INPUT HOPPER/
013663      100
013664      122
013665      105
013666      115
013667      117
013670      126
013671      105
013672      040
013673      101
013674      114
013675      114
013676      040
013677      103
013700      101
013701      122
013702      104
013703      123
013704      040
013705      106
013706      122
013707      117
013710      115
```

```
013711      040
013712      124
013713      110
013714      105
013715      040
013716      111
013717      116
013720      120
013721      125
013722      124
013723      040
013724      110
013725      117
013726      120
013727      120
013730      105
013731      122
013732      057

013733      057                 MSG61:  .ASCII  /?RESTORE CARDS IN THE INPUT HOPPER/<1
013734      100
013735      122
013736      105
013737      123
013740      124
013741      117
013742      122
013743      105
013744      040
013745      103
013746      101
013747      122
013750      104
013751      123
013752      040
013753      111
013754      116
013755      040
013756      124
013757      110
013760      105
013761      040
013762      111
013763      116
013764      120
013765      125
013766      124
013767      040
013770      110
013771      117
013772      120
013773      120
013774      105
013775      122
013776      057
```

```
013777     057                 MSG7:   .ASCII  |/@RAISE OUTPUT STACKER PRESSURE ARM SLIGHTLY ABOVE HORIZONTAL @ THEN LOWER
IT/|
014000     100
014001     122
014002     101
014003     111
014004     123
014005     105
014006     040
014007     117
014010     125
014011     124
014012     120
014013     125
014014     124
014015     040
014016     123
014017     124
014020     131
014021     103
014022     113
014023     105
014024     122
014025     040
014026     120
014027     122
014030     105
014031     123
014032     123
014033     125
014034     122
014035     105
014036     040
014037     101
014040     122
014041     115
014042     040
014043     123
014044     114
014045     111
014046     107
014047     110
014050     124
014051     114
014052     131
014053     040
014054     101
014055     102
014056     117
014057     126
014060     105
014061     040
014062     110
014063     117
014064     122
```

```
014065      111
014066      132
014067      117
014070      116
014071      124
014072      101
014073      114
014074      040
014075      100
014076      040
014077      124
014100      110
014101      105
014102      116
014103      040
014104      114
014105      117
014106      127
014107      105
014110      122
014111      040
014112      111
014113      124
014114      057

014115      057                 MSG7A:  .ASCII  /?LOWER OUTPUT STACKER PLATE TO BOTTOM/<15>
014116      100
014117      114
014120      117
014121      127
014122      105
014123      122
014124      040
014125      117
014126      125
014127      124
014130      120
014131      125
014132      124
014133      040
014134      123
014135      124
014136      101
014137      103
014140      113
014141      105
014142      122
014143      040
014144      120
014145      114
014146      101
014147      124
014150      105
014151      040
014152      124
```

```
014153      117
014154      040
014155      102
014156      117
014157      124
014160      124
014161      117
014162      115
014163      057

014164      057                 MSG8:   .ASCII  /?HOLD DOWN THE SWITCH AT THE BOTTOM OF THE INPUT HOPPER?/
014165      100
014166      110
014167      117
014170      114
014171      104
014172      040
014173      104
014174      117
014175      127
014176      116
014177      040
014200      124
014201      110
014202      105
014203      040
014204      123
014205      127
014206      111
014207      124
014210      103
014211      110
014212      040
014213      101
014214      124
014215      040
014216      124
014217      110
014220      105
014221      040
014222      102
014223      117
014224      124
014225      124
014226      117
014227      115
014230      040
014231      117
014232      106
014233      040
014234      124
014235      110
014236      105
014237      040
014240      111
```

```
014241      116
014242      120
014243      125
014244      124
014245      040
014246      110
014247      117
014250      120
014251      120
014252      105
014253      122
014254      057

014255      057                 MSG8A:  .ASCII  /?@LIFT SWITCH UNDER RIFFLE CAP?/
014256      100
014257      114
014260      111
014261      106
014262      124
014263      040
014264      123
014265      127
014266      111
014267      124
014270      103
014271      110
014272      040
014273      125
014274      116
014275      104
014276      105
014277      122
014300      040
014301      122
014302      111
014303      106
014304      106
014305      114
014306      105
014307      040
014310      103
014311      101
014312      120
014313      057

014314      057                 MSG9:   .ASCII  /?@BLOCK THE CARD READER STATION TO PREVENT A CARD GOING THRU, AND?/
014315      100
014316      102
014317      114
014320      117
014321      103
014322      113
014323      040
014324      124
014325      110
```

```
014326      105
014327      040
014330      123
014331      101
014332      122
014333      104
014334      040
014335      122
014336      105
014337      121
014340      124
014341      105
014342      122
014343      040
014344      123
014345      124
014346      101
014347      124
014350      111
014351      117
014352      116
014353      040
014354      124
014355      117
014356      040
014357      120
014360      122
014361      105
014362      126
014363      105
014364      116
014365      124
014366      040
014367      121
014370      040
014371      123
014372      121
014373      122
014374      104
014375      040
014376      127
014377      117
014400      111
014401      116
014402      107
014403      040
014404      124
014405      110
014406      122
014407      125
014410      054
014411      040
014412      101
014413      116
014414      124
```

```
014415      057

014416      057                 MSG10:  .ASCII  /?REMOVE JAMMED CARD/<
014417      120
014420      122
014421      105
014422      115
014423      117
014424      126
014425      105
014426      040
014427      112
014430      101
014431      115
014432      115
014433      105
014434      104
014435      040
014436      103
014437      101
014440      122
014441      104
014442      057

014443      057                 MSG11:  .ASCII  /?HOLD THE OUTPUT STACKER GATE OPEN, THEN/<
014444      120
014445      110
014446      117
014447      114
014450      104
014451      040
014452      124
014453      110
014454      105
014455      040
014456      117
014457      125
014460      124
014461      120
014462      125
014463      124
014464      040
014465      123
014466      124
014467      101
014470      103
014471      113
014472      105
014473      122
014474      040
014475      127
014476      101
014477      124
014500      105
014501      040
```

```
014502      117
014503      120
014504      105
014505      116
014506      056
014507      040
014510      124
014511      110
014512      105
014513      116
014514      057

014515      257                 MSG12:  .ASCII  /?PLACE SPECIAL DARK-LIGHT CHECK CARDS (SEE LISTING, TESTG)/
014516      120
014517      120
014520      114
014521      101
014522      103
014523      105
014524      040
014525      123
014526      120
014527      105
014530      103
014531      111
014532      101
014533      114
014534      040
014535      124
014536      101
014537      122
014540      113
014541      055
014542      114
014543      111
014544      107
014545      110
014546      124
014547      040
014552      123
014551      112
014552      105
014553      103
014554      113
014555      040
014556      103
014557      101
014562      122
014561      104
014562      123
014563      040
014564      052
014565      123
014566      105
014567      105
```

```
014570  040
014571  114
014572  111
014573  123
014574  124
014575  111
014576  116
014577  107
014600  054
014601  040
014602  124
014603  125
014604  123
014605  124
014606  107
014607  051

014610  170                     ,ASCII  /@AT THE BOTTOM OF THE INPUT STACK/
014611  131
014612  124
014613  040
014614  124
014615  110
014616  105
014617  040
014620  102
014621  117
014622  124
014623  124
014624  117
014625  115
014626  040
014627  117
014630  106
014631  040
014632  124
014633  110
014634  105
014635  040
014636  111
014637  116
014640  120
014641  125
014642  124
014643  040
014644  123
014645  124
014646  101
014647  123
014650  113
014651  057

014652  057                     MSG131  ,ASCII  /-@DECK    CARD  COLUMN PATTERN READ1 READ2  CODED  READ-/
014653  100
014654  124
```

```
014655      105
014656      103
014657      113
014660      040
014661      040
014662      040
014663      040
014664      103
014665      101
014666      122
014667      104
014670      040
014671      040
014672      103
014673      117
014674      114
014675      125
014676      115
014677      116
014700      040
014701      120
014702      101
014703      124
014704      124
014705      105
014706      122
014707      116
014710      040
014711      122
014712      105
014713      101
014714      104
014715      061
014716      040
014717      122
014720      105
014721      101
014722      104
014723      062
014724      040
014725      040
014726      103
014727      117
014730      104
014731      105
014732      104
014733      040
014734      040
014735      122
014736      105
014737      101
014740      104
014741      057

014742      057                 MSG141  ,ASCII  !/@ALPHA /!
```

```
014743      100
014744      101
014745      114
014746      120
014747      110
014750      101
014751      040
014752      057

014753      057                 MSG15: .ASCII  /?BINARY/
014754      120
014755      102
014756      111
014757      116
014760      101
014761      122
014762      131
014763      057

014764      057                 MSG16: .ASCII  /?BIT 15 WAS SET/
014765      120
014766      102
014767      111
014770      124
014771      040
014772      061
014773      065
014774      040
014775      127
014776      101
014777      123
015000      040
015001      123
015002      105
015003      124
015004      057

015005      057                 MSG17: .ASCII  /?REMEDY THE ERROR CONDITION AND PRESS 'CONTINUE'?/
015006      120
015007      122
015010      105
015011      115
015012      105
015013      104
015014      131
015015      040
015016      124
015017      110
015020      105
015021      040
015022      105
015023      122
015024      122
015025      117
015026      122
```

```
015027      040
015030      103
015031      117
015032      116
015033      104
015034      111
015035      124
015036      111
015037      117
015040      116
015041      040
015042      101
015043      116
015044      104
015045      040
015046      120
015047      122
015050      105
015051      123
015052      123
015053      040
015054      047
015055      103
015056      117
015057      116
015060      124
015061      111
015062      116
015063      125
015064      105
015065      047
015066      100
015067      057

015070      057                 MSG18:  .ASCII  /\@BIT 8 WAS SET\/
015071      120
015072      102
015073      111
015074      124
015075      040
015076      070
015077      040
015100      127
015101      101
015102      123
015103      040
015104      123
015105      105
015106      124
015107      057

015110      057                 MSG19:  .ASCII  /\@COLUMN READ1 READ2  CARDS ERRORS\/
015111      120
015112      103
015113      117
```

```
015114      114
015115      125
015116      115
015117      116
015120      040
015121      122
015122      125
015123      131
015124      134
015125      061
015126      040
015127      122
015130      125
015131      131
015132      134
015133      062
015134      040
015135      040
015136      103
015137      101
015140      122
015141      134
015142      123
015143      040
015144      125
015145      122
015146      122
015147      117
015150      122
015151      123
015152      057

        000021                      .END
```

```
ADINT   000000R     CONTD   005450     LOOP4   001262     RETURN  012002
ALCNT   006620      CONTG   010374     LOOP4B  001300     SAVPC   011564
ALCNT1  006702      COUNT   000001R    LOOP5   001412     SAVPSR  011566
ALCNT2  006714      CRB1    000004R    LOOP6   001550     SAVR2   011556
ALLOON  006566      CRB2    000620     LOOP7   001702     SAVR3   011560
ALLOK   006742      CRLF    012074     LOOP8   002074     SAVR4   011562
ALP1    005426      CRLF4   012126     LOOPF   010262     SCONT1  005610
ALPCD   012150      CRS     000003R    LOOPG   010414     SCOPE   104400
ALPEND  012646      D,1     011752     MSG1    013350     SCOPEC  011716
ALWAIT  006776      D,2     011764     MSG10   014416     SET1    004466
B,CK    011544      DAT1    006556     MSG11   014443     SET14   003204
BEGIN   000674      DAT2    006560     MSG12   014515     SET2    004254
BELL    011322      DATENC  006562     MSG13   014652     SET3    004042
BINCD   012650      DATST   005352     MSG14   014742     SET4    003630
BINEND  013346      DATST1  005362     MSG15   014753     SET5    003416
C,CONT  011660      DATST2  005366     MSG16   014764     SET7    003032
C,WAIT  011624      DBRCK8  002156     MSG17   015005     SETUP   000632
CARDIM  011272      DCNT1   005504     MSG18   015070     SP      000006R
CDCNT   006552      DECK    006544     MSG19   015110     SPACE   011402
CK4     001314      DECKCK  006420     MSG1A   013433     SPACEX  011432
CK5     001460      DONE3   001240     MSG2    013467     SR      177570
CK9     002276      DONE7   001754     MSG3    013534     SRVC    005544
CKBIT8  011346      END     006402     MSG3A   013574     SRVC1   006774
CKCOL   011034      END24   005304     MSG4    013627     SRVC2   007036
CKCRD   011020      ENDCK   005324     MSG5    013662     SRVC2A  007044
CKDONE  011256      ER1     005742     MSG6    013733     STACK   000600
CKFAIL  011112      ERCM11  007112     MSG7    013777     T2INT   004744
CKHLT   011244      ERCR11  007104     MSG7A   014115     T2INTA  004770
CKLOOP  010734      ERFLG   000630     MSG8    014164     TCSR    000606
CKLP1   010754      ERR6    001652     MSG8A   014255     TDBR    000610
CKLP2   011060      ERSET   005710     MSG9    014314     TEND    006550
CKNOHD  011150      FAIL    005770     NOHD    006120     TEST    000736
CKSAME  010676      FAILC   006030     NOP     000240     TEST1   000744
CKSIT   011074      FAILC1  006312     NOTCD   006514     TEST10  002326
CKSIT1  011010      FAILCN  006076     NOTCD1  006536     TEST11  002450
CLCNT   006554      FLAG    000612     NOTCOL  006316     TEST12  002552
CNT23A  005034      HLT     104000     NOTRP   000710     TEST13  002632
CNT23B  005052      INIT    011274     NOTRP1  005456     TEST14  003044
CNT23C  005072      INTFLG  000602     NXCRD   006444     TEST15  003216
CNT23D  005110      INTVC   000604     OFF6    001640     TEST16  003430
CNT23E  005130      ITCNT   012000     PC      000007R    TEST17  003642
CNT23F  005150      ITMAX   011776     PRINT   011434     TEST18  004054
CNT23G  005174      KCRB1   000616     PROC    000626     TEST19  004266
CNT24A  005242      KCRS    000614     PROCT   011570     TEST1A  000746
CNT24B  005272      L,CNT   012042     PSR     177776     TEST2   001042
CONT10  002440      L,EOMK  012072     PTOFF   006564     TEST20  004500
CONT11  002542      L,INC   012016     R2      000002R    TEST21  004604
CONT2   001152      L,TOUT  012020     R5      000005R    TEST22  004700
CONT21  004670      LASTCD  006354     REST8   002214     TEST23  005006
CONT22  004774      LASTCK  006336     REST9   002320     TEST24  005212
CONT4   001334      LOGIC   006372     RESTRT  000702     TEST3   001166
CONT8   002166      LOOP3   001212     RETRNX  010674     TEST4   001250
```

```
TEST5   001400
TEST6   001534
TEST7   001666
TEST8   002052
TEST9   002222
TESTA   007132
TESTB   007270
TESTC   007420
TESTD   007552
TESTE   010026
TESTF   010174
TESTG   010342
TESTX   010574
TIM24   005276
TINT10  002410
TINT11  002530
TINT12  002614
TINT13  002750
TINT14  003142
TINT15  003334
TINT16  003546
TINT17  003760
TINT18  004172
TINT19  004404
TINT20  004560
TNINT   004662
TOTCRD  011270
TOTERR  011266
TOUT    012004
TRAPX   000720
TRFLG   000624
TRP1    005466
TRTRAP  000622
TSTA    007120
TSTART  006546
WAIT9   002260
XLOOP   010664

ERRORS DETECTED: 0

RUN-TIME:  25 SECONDS

5K CORE USED
```

DEC-11-NIZB-D

PDP-11

DEVICE DRIVER PACKAGE

FEBRUARY 1972

COPYRIGHT © 1971, 1972 BY DIGITAL EQUIPMENT CORPORATION

SOFTWARE SUPPORT CATEGORY

The software described in this document is supported by DEC under Category I, as defined on page ii of this document.

THIS DOCUMENT IS FOR INFORMATION PURPOSES ONLY AND IS SUBJECT TO CHANGE WITHOUT NOTICE.

pdp11 PROGRAM LIBRARY

SOFTWARE SUPPORT CATEGORIES

Digital Equipment Corporation (DEC) makes available four categories of software. These categories reflect the types of support a customer may expect from DEC for a specified software product. DEC reserves the right to change the category of a software product at any time. The four categories are as follows:

CATEGORY I
Software Products Supported at no Charge

This classification includes current versions of monitors, programming languages, and support programs provided by DEC. DEC will provide installation (when applicable), advisory, and remedial support at no charge. These services are limited to original purchasers of DEC computer systems who have the requisite DEC equipment and software products.

At the option of DEC, a software product may be recategorized from Category I to Category II for a particular customer if the software product has been modified by the customer or a third party.

CATEGORY II
Software Products that Receive Support for a Fee

This category includes prior versions of Category I programs and all other programs available from DEC for which support is given. Programming assistance (additional support), as available, will be provided on these DEC programs and non-DEC programs when used in conjunction with these DEC programs and equipment supplied by DEC.

CATEGORY III
Pre-Release Software

DEC may elect to release certain software products to customers in order to facilitate final testing and/or customer familiarization. In this event, DEC will limit the use of such pre-release software to internal, non-competitive applications. Category III software is only supported by DEC where this support is consistent with evaluation of the software product. While DEC will be grateful for the reporting of any criticism and suggestions pertaining to a pre-release, there exists no commitment to respond to these reports.

CATEGORY IV
Non-Supported Software

This category includes all programs for which no support is given

CONTENTS

		Page
CHAPTER 1.	INTRODUCTION	1-1
CHAPTER 2.	DRIVER FORMAT	2-1
2.1	Structure	2-1
2.1.1	Driver Interface Table	2-1
2.1.2	Set-up Routines	2-1
2.1.3	Interrupt Servicing	2-1
2.1.4	Error Handling	2-2
2.2	Interface to the Driver	2-2
2.2.1	Control Interface	2-2
2.2.2	Interrupt Interface	2-2
CHAPTER 3.	STAND-ALONE USAGE	3-1
3.1	Driver Assembled with Program	3-1
3.1.1	Setting Interrupt Vector	3-1
3.1.2	Parameter Table for Driver Call	3-1
3.1.3	Calling the Driver	3-2
3.1.4	User Registers	3-3
3.1.5	Return from Driver	3-4
3.1.6	Irrecoverable Errors	3-5
3.1.7	General Comment	3-6
3.2	Drivers Assembled Separately	3-6
3.3	Device-independent Usage	3-8
APPENDIX A	I-O DRIVERS WITHIN THE DISK OPERATING SYSTEM	A-1
A.1	Driver Structure	A-1
A.2	Monitor Calling	A-2
A.3	Driver Routines	A-4
A.3.1	TRANSFER	A-4
A.3.2	Interrupt Servicing	A-4
A.3.3	OPEN	A-5
A.3.4	CLOSE	A-7
A.3.5	SPECIAL	A-7
A.4	Drivers for Terminals	A-7

CHAPTER 1

INTRODUCTION

Subroutines to handle I/O transfers between a PDP-11 and each of its peripheral devices are developed as required for use within the Disk Operating System (DOS). These subroutines are made available within an I/O Utilities Package for the benefit of PDP-11 users who have configurations unable to support DOS or who wish to run programs outside DOS control.

All the subroutines associated with one peripheral device together form an entity which is known as a Driver. The purpose of this document is to provide a general description of a driver and to show how it may be used in a stand-alone environment. The unique properties of each driver are discussed in separate documents issued as supplements to this one. The I/O Utilities Package for any system is determined by the peripherals of that system. Thus the full documentation for a particular package consists of this document and the applicable supplements.

Within this document, Chapter 2 consists of an outline of the established driver structure and its interface to the program using it. Chapter 3 then illustrates how a stand-alone program can match this interface in order to make immediate use of each driver as supplied within the package. For the benefit of those users who require a more detailed description of the driver format, perhaps so that they can write their own drivers for other unsupported devices in a similar fashion, the standard specification for DOS driver has been attached as Appendix A. It is assumed that the reader is familiar with the basic hardware concepts of the PDP-11 as described in the PDP-11 Processor Handbook and with the Paper Tape Software as described in the Programming Handbook (DEC-11-GGPC-D).

%%%%%%

CHAPTER 2

DRIVER FORMAT

2.1 Structure

The basic principle of all drivers under the DOS Monitor is that they must present a common interface to the routines using them in order to provide for device-independent operation. The subroutines are structured to meet this end. Moreover, the driver may be loaded anywhere in memory under Monitor control. Its code must always, therefore, be position-independent.

The detailed description of a driver is found in Appendix A. This chapter is concerned with driver interfaces.

2.1.1 Driver Interface Table

The first section of each driver consists of a table which contains, in a standard format, information on the nature and capabilities of the device it represents and entry pointers to each of its subroutines. The calling program may then use this table as required, regardless of the device being called.

2.1.2 Setup Routines

Each driver is expected to handle its device under the PDP-11 interrupt system. When called by a program, therefore, a driver subroutine merely initiates the action required by setting the device hardware registers appropriately. It then returns to the calling program by a standard subroutine exit.

The main setup routine prepares for a data transfer to or from the device, using parameters supplied by the calling program. Normally, blocks of data will be moved at each transfer. The driver will only return control to the program when the whole block has been actioned or when it is unable to continue because there is no more data available.

The driver may also contain subroutines by which the calling program may request start-up or shut-down action, such as leader or trailer code at a paper-tape punch, or some special function provided by the device hardware (or a software simulation of that for some similar device), e.g., "rewind" of a magnetic tape (or DECtape).

2.1.3 Interrupt Servicing

The nature of the driver routine to service device interrupts is

particularly dependent upon the extent of the hardware provisions of the device for controlling transfers. In general, the driver determines the cause of the interrupt and checks whether the last action was performed correctly or was prevented by some error condition. If more device action is needed to satisfy the program, the driver again initiates that action and takes a normal interrupt exit. If the program request has been fully met, control is returned to the program at an address supplied at the time of the call.

2.1.4 Error Handling

Device errors may be handled in two ways. There are some errors for which recovery can be programmed; the driver will, if appropriate, attempt this itself (as in the case of parity or timing failure on a bulk-storage device) or will recall the program with the error condition flagged (as at the end of a physical paper tape). Other errors will normally require action externally, perhaps by an operator. For the latter, the driver calls a common error handler based on location 34 (IOT call) with supporting information on the processor stack.

2.2 Interface to the Driver

2.2.1 Control Interface

The principal link between a calling program and any driver subroutine is the first word of the driver table. In order to provide the control parameters for a device operation, the calling program prepares a list in a standardized form and places a pointer to the list in the driver link. The called driver then uses the pointer to access the parameters. If the driver need then return status information, it may again place this in the list area via the link-word.

The first word of the driver also may act as an indicator in that while it remains Ø, the driver is not already busy upon some task, whereas when the word contains a list-pointer, the driver is assumed to be busy. Since most drivers can only support one job at a time, the link-word state can be significant.

2.2.2 Interrupt Interface

Although the driver will always expect to use the interrupt system, it does not itself ensure that its interrupt vector in the memory area below 400 has been set up correctly; the Monitor under DOS takes care of this. However, the Driver Table contains the necessary information to allow the vector to be set correctly.

CHAPTER 3

STAND-ALONE USAGE

Because each driver is designed for operation within the device-independent framework of DOS Monitor, it may be similarly used in other applications. Possible methods will be discussed later. However, since the easiest way to use the driver is to assemble it with the program requiring it, this will be described first.

3.1 Driver Assembled with Program

3.1.1 Setting Interrupt Vector

As noted in Section 2.2.2, the calling program must first correctly set the device transfer vector within memory locations 0-377. The address of the driver's interrupt entry point can be identified on the source listing by the symbolic name which appears as the content of the Driver Table Byte, DRIVER+5. The priority level at which the driver expects to process the interrupt is at byte DRIVER+6. For a program which can use position-dependent code, the setup sequence may be:

```
MOV   #DVRINT, VECTOR      ;SET INT. ADDRESS
MOVB  DRIVER+6, VECTOR+2   ;SET PRIORITY
CLRB  VECTOR+3             ;CLEAR UPPER STATUS BYTE
```

(where the Driver Table shows at DRIVER +5: .BYTE DVRINT-DRIVER).

If the program must be position-independent, it may take advantage of the fact that the Interrupt Entry address is actually stored as an offset from the start of the driver, as illustrated above. In this case, a sample sequence might be:

```
MOV   PC,R1                ;GET DRIVER START
ADD   #DRIVER-.,R1
MOV   #VECTOR,R2           ;...& VECTOR ADDRESSED
CLR   @R2                  ;SET INT. ADDRESS
MOVB  5(R1),@R2            ;...AS START ADDRESS+OFFSET
ADD   R1,(R2)+
CLR   @R2                  ;SET PRIORITY
MOVB  6(R1),@R2
```

3.1.2 Parameter Table for Driver Call

For any call to the driver, the program must provide the list of

control arguments mentioned in Section 2.2.1. This list must adhere in general to the following format:[1]

```
[SPECIAL FUNCTION CODE][2]
[BLOCK NO.][3]
STARTING MEMORY ADDRESS FOR TRANSFER
NO. OF WORDS to be transferred (2's complement)
STATUS CONTROL showing in Bits:
  Ø-2:  Function (octally 2=WRITE, 4=READ)[4]
  8-1Ø: Unit (if Device can consist of several, e.g., DECtape)
  11:   Direction for DECtape travel (Ø = Forward)
ADDRESS for RETURN ON COMPLETION
[RESERVED FOR DRIVER USE][5]
```

The list itself may be assembled into the required format if its content will not vary. The driver may return information in the area as described in a later paragraph; however, this will not corrupt the program data and it is removed by the driver before it begins its next operation.

On the other hand, most programs will probably wish to use the same area for the lists for several tasks or even between different drivers. In this case, the program must contain the necessary routine to set up the list for each task before making the driver call, perhaps as illustrated in the next paragraph. It must be noted, however, that the driver may wish to refer to the list again when it is recalled by an interrupt or to return information to the calling program. Therefore, the list must not be changed until any driver has completed a function requested; for concurrent operations, different list areas must be provided.

3.1.3 Calling the Driver

To enable the driver to access the parameter list, the program must set the first word of the driver to an address six bytes less than that

---

[1] In some cases, it may be further extended as discussed in later paragraphs.

[2] Required only if Driver is being called for Special Function.

[3] Required only if the Device is bulk storage (e.g., Disk or DECtape).

[4] Most devices transfer words regardless of their content, i.e. ASCII or Binary. Some devices, e.g., Card Reader, may be handled differently for the two modes; for these, Bit Ø must also be set to indicate ASCII=Ø, Binary=1. (In these cases, the driver always produces or accepts ASCII even though the device itself uses some other code.)

[5] This word may be omitted if the device is bulk storage (see below).

of the word containing MEMORY START ADDRESS. It may then call the driver subroutine required directly by a normal JSR PC,xxxx call.

As an example, the following position-independent code might appear in a program which wishes to read Blocks #1ØØ-1Ø3 backward from DECtape Unit into a buffer starting at address BUFFER:

```
        MOV   PC,RØ                    ;GET TABLE ADDRESS
        ADD   #TABLE+12-.,RØ
        MOV   PC,@RØ                   ;GET & STORE...
        ADD   #RETURN-.,@RØ            ;...RETURN ADDRESS
        MOV   #54Ø4,-(RØ)              ;SET READ REV. UNIT 3
        MOV   #-1Ø24.,-(RØ)            ;4 BLOCKS REQUIRED
        MOV   PC,-(RØ)                 ;GET & STORE
        ADD   #BUFFER-.,@RØ            ;...BUFFER ADDRESS
        MOV   #1Ø3,-(RØ)               ;START BLOCK
        CMP   -(RØ),-(RØ)              ;SUBTRACT 4 FROM POINTER
        MOV   RØ,DT                    ;SET DRIVER LINK
        JSR   PC,DT.TFR                ;GOTO TRANSFER ROUTINE
WAIT:   .                              ;RETURNS HERE WHEN
        :                              ;...TRANSFER UNDERWAY
RETURN: .                              ;RETURNS HERE WHEN
        :                              ;... TRANSFER COMPLETE
TABLE:  .WORD Ø                        ;LIST AREA SET
        .WORD Ø                        ;...BY ABOVE SEQUENCE
        .WORD Ø
        .WORD Ø
        .WORD Ø
```

3.1.4 User Registers

During its setup operations for the function requested, the driver assumes that Processor Registers Ø-5 are freely available for its purpose. If their contents are of value, the program must save them before the driver is called.

While servicing intermediate interrupts, the driver may need to save or restore these registers. It expects to have available two subroutines for the purpose (provided by the Monitor under DOS) It accesses them via addresses in memory locations 44 (SAVE) and 46 (RESTORE) using the sequence:

```
MOV   @#44,-(SP)          ;OR 'MOV   @#46,-(SP)
JSR   R5,@(SP)+
```

The program must, therefore, contain these subroutines. They might, for example, be as follows:

```
SAVE:       MOV   R4,-(SP)      ;SAVE RØ-4
            MOV   R3,-(SP)      ;...R5 SAVED BY CALL
            MOV   R2,-(SP)
            MOV   R1,-(SP)
            MOV   RØ,-(SP)
            MOV   R5,PC         ;EXIT TO CALLER

RESTOR:     INC   (SP)+         ;FORGET CALL R5
            MOV   (SP)+,RØ      ;RESTORE RØ-4
            MOV   (SP)+,R1
            MOV   (SP)+,R2
            MOV   (SP)+,R3
            MOV   (SP)+,R4
            RTS   R5            ;R5 RESET ON EXIT
```

It must also ensure that their start addresses are set into the correct locations.

At its final interrupt, the driver always saves the contents of Registers Ø-5 before returning control to the calling program completion return.

3.1.5 Returns from Driver

As shown in the example in section 3.1.3, the driver returns control to the calling program immediately after the JSR as soon as it has set the device in motion. The program may then wait or carry out some alternative operations until the driver signals completion by returning at the address supplied, i.e., RETURN above. Prior to this, the program should not attempt to access the data being read in, or to refill a buffer being written out.

The program routine beginning at address RETURN will vary according to the device in use. In general, the driver has given control to the routine for one of two reasons, namely, the function has been satisfactorily performed, or it cannot be carried out due to some hardware failure with which the driver is unable to cope, though the program may. If the latter, the driver uses the STATUS word in the program list to show the cause:

Bit 15 = 1	indicates that a device parity or timing failure has occurred and the driver has not been able to overcome this, perhaps after several attempts.
Bit 14 = 1	shows that the end of the data available has been reached.

The driver places in RØ the content of its first word as a pointer to the list concerned.

In addition, the driver may have transferred only some of the data required. In this case, it will show, in the RESERVED word of the program list, a negative count of the words not transferred in addition to setting Bit 14 of the STATUS. As mentioned in the note in Section 3.1.2, this applies only to non-bulk storage devices. The drivers for DECtape or Disks[1] always endeavor to complete the full transfer, even beyond a parity failure, or they take more drastic action (see Section 3.1.6).

It is thus the responsibility of the program RETURN routine to check the information supplied by the driver in order to verify that the transfer was satisfactory and to handle the error situations accordingly.

In addition, the routine must contain a sequence to take care of the Processor Stack, Registers, etc. As noted earlier, the driver takes the completion return address after an interrupt and has saved Registers 0-5 on the stack above the Interrupt Return Address and Status. The program routine should, therefore, contain some sequence to restore the processor to its state prior to such interrupt, e.g., using the same Restore subroutine illustrated earlier:

```
MOV   @#46,-(SP)          ;CALL REGISTER RESTORE
JSR   R5,@(SP)+
  .
  .
  .
RTI                       ;RETURN TO INTERRUPTED PROG.
```

3.1.6 Irrecoverable Errors

All hardware errors other than those noted in the previous paragraph are more serious in that they cannot normally be overcome by the program or the driver on its behalf. Some of these could be due to an operator fault, such as an omission to turn a paper tape reader on or to set the correct unit number on a DECtape transport. Once the operator has rectified the problem, the program could continue. Other errors, however, will require hardware repair or even software repair, e.g., if the program asks for Block 2000 on a device having a maximum of 1000. In general, all these errors will result in the driver placing identifying information on the processor stack and calling IOT to produce a trap through location 34.

[1]This includes RF11 Disk: although this is basically word-oriented, it is assumed to be subdivided into 64-word blocks.

Under DOS, the Monitor provides a routine which prints a teleprinter message when this occurs. In a stand-alone environment, the program using the driver must itself contain the routine to handle the trap (unless the user wishes to modify the driver error exits before assembly). The handler format will depend upon the program. Should it wish to take advantage of the information supplied by the driver, the format is as follows:

(SP):	Return Address }	Stored by IOT Call
2(SP):	Return Status }	
4(SP):	Error No. Code	generally unique to driver
5(SP):	Error Type Code:	1 = Recoverable after Operator Action 3 = No recovery
6(SP):	Additional Information	such as content of Driver, Control Register, Driver Identity, etc.

As a rule, the driver will expect a return following the IOT call in the case of errors in Type 1 but will contain no provision following a return from Type 3.

3.1.7 General Comment

The source language of each driver has been written for use with the DOS version of the Assembler which requires certain statements which will not be accepted by the Paper Tape Software PAL-11A, in particular: .TITLE & .GLOBL. These should be edited out before the source is used. Similarly, an entry in the driver table gives the device name as .RAD5Ø 'DT' to obtain a specially packed format used internally by DOS. If the user still wishes to keep the name, for instance for identification purposes as discussed in section 3.3, .RAD5Ø might easily be changed to .ASCII without detrimental effect, or it can be replaced with .WORD Ø .

3.2 Drivers Assembled Separately

Rather than assemble the driver with every program requiring its availability, the user may wish to hold it in binary form and attach it to the program only when loaded. This is readily possible; the only requirement is that the start address of the driver should be known or can be determined by the program.

The example in section 3.1.2 showed that the Interrupt Servicing routine can be accessed through an offset stored in the Driver Table. The same technique can be used to call the setup subroutines, as these also have corresponding offsets in the Table, as follows:

```
DRIVER+7     Open[1]
      +10    Transfer
      +11    Close[1]
      +12    Special Functions[1]
```

The problem, of course, is the start address. There is always the obvious solution, that of assembling the driver at a fixed location so that each program using it can immediately reference the location chosen. This, however, ceases to be convenient when the program itself has to avoid the area given to the driver. A more general method is to relocate the driver as dictated by the program using it, thus taking advantage of the position-independent nature of the driver. The Absolute Loader, described in the Paper Tape Software Handbook (DEC-11-GGPB-D), Chapter 6, provides the capability of continuing a load from the point at which it ended. Using this facility to enter the driver immediately after the program, the program itself might contain the following code to call the subroutine to perform the transfer illustrated in section 3.1.3:

```
        MOV    PC,R1              ;GET DRIVER START ADDRESS
        ADD    #PRGEND-.,R1
        MOV    PC,RØ              ;GET TABLE ADDRESS
        ADD    #TABLE+12-.,RØ     ;& SET UP AS SHOWN
          .                       ;...IN SECTION 3.1.3
          .
          .
        CMP    -(RØ),-(RØ)        ;FINAL POINTER ADJUSTMENT
        MOV    RØ,@R1             ;STORE IN DRIVER LINK
        CLR    -(SP)              ;GET BYTE SHOWING...
        MOVB   1Ø(R1),@SP         ;...TRANSFER OFFSET
        ADD    (SP)+,R1           ;COMPUTE ADDRESS
        JSR    PC,@R1             ;GO TO DRIVER
          .
          .
          .
PRGEND:
       .END
```

This technique may be extended to cover situations in which several drivers are used by the same program, provided that it takes account of the size of each driver (this being already known because of prior assembly) and that the drivers themselves are always loaded in the same order.

For example, to access the second driver, the above sequence would be modified to:

[1]If the routine is not provided, these are Ø.

```
        MOV    PC,R1            ;GET DRIVER 1 ADDRESS
        ADD    #PRGEND-.,R1
        ADD    #DVR1SZ,R1       ;STEP TO DRIVER 2
          .
          .
          .
DVR1SZ=
PRGEND:
       .END
```

An alternative method may be to use the Relocatable Assembler PAL-11S in association with the Linker program LINK-11S, both of which are available through the DECUS Library. The start address of each driver is identified as a global. Any calling program need, therefore, merely include a corresponding .GLOBL statement, e.g., .GLOBL DT.

3.3 Device-independent Usage

As mentioned earlier, the drivers are designed for use in a device-independent environment, i.e., one in which a calling program need not know in advance which driver has been associated with a table for a particular execution run. One application of this type might be to allow line-printer output to be diverted to some other output medium because the line-printer itself is currently not available. Another might be to provide a general program to analyze data samples although these on one occasion might come directly from an Analog to Digital converter and on another be stored on a DECtape, because the sampling rate was too high to allow immediate evaluation.

As a rule, programs of this type should be written to cater for all the facilities that any one device might offer, but not necessarily all of them. For instance, the program should ask for start-up procedures because it may sometime use a paper tape punch which provides them, even though it may normally use DECtape which does not. As noted in section 2.1.1, the driver table contains an indication of its capabilities to cater for this situation. The program can thus examine the appropriate item before calling the driver to perform some action. As an example, the code to request start-up procedures might be (assuming RØ already set to List Address):

```
MOV    #DVRADD,R1       ;GET DRIVER ADDRESS
TSTB    2(R1)           ;BIT 7 SHOWS...
BPL    NOOPEN           ;...OPEN ROUTINE PRESENT
MOV    RØ,@R1           ;STORE TABLE ADDRESS
CLRB   -(SP)            ;BUILD ADDRESS
MOVB   7(R1),@SP        ;...OF THIS ROUTINE
ADD    (SP)+,R1
```

```
        JSR   PC,@R1            ;...& GO TO IT
                                ;FOLLOWED POSSIBLY BY
                                ;WAIT AND COMPLETION
                                ;PROCESSING
NOOPEN:                         ;RETURN TO COMMON OPERATION
```

Similarly, the indicators show whether the device is capable of performing input or output or both, whether it can handle ASCII data or Binary data, whether it is a bulk storage device capable of supporting a directory structure or is a terminal-type device requiring special treatment and so on. Other table entries show the device name as identification and how many words it might normally expect to transfer at a time (in 16-word units). All of the information may readily be examined by the calling program, thus enabling the use perhaps of a common call sequence for any I/O operation, as for example:

```
        MOV   #DVRADR, R5       ;SET DRIVER START
        JSR   R5, IOSUB         ;CALL SET UP SUB
        BR    WAIT              ;SKIP TABLE FOLLOWING ON RETURN
        .WORD 1Ø                ;TRANSFER REQUIRED
        .WORD 1Ø3               ;BLOCK NO.
        .WORD BUFFER            ;BUFFER ADDRESS
        .WORD -256.             ;WORD COUNT
        .WORD 4Ø4               ;READ FROM UNIT 1
        .WORD RETURN            ;EXIT ON COMPLETION
        .WORD Ø                 ;RESERVED
WAIT:                           ;CONTINUE HERE...
        .                       ;WHILE TRANSFER IN PROGRESS
        .
        .
        .

IOSUB:  MOV   @SP,RØ            ;PICK UP DRIVER ADDR
        MOV   R5,R1             ;SET POINTER TO LIST
        TST   (R1)+             ;BUMP TO COLLECT CONTENT
        .                       ;ROUTINE CHECKS ON DEVICE..
        .                       ;..CAPABILITY USING R1
        .                       ;..TO ACCESS LIST &
        .                       ;..RØ THE DRIVER TABLE
        .                       ;IF O.K...
        MOV   @R1,R1            ;GET ROUTINE OFFSET
        ADD   RØ,R1
        CLR   -(SP)             ;USE IT TO BUILD
        MOVB  @R1,@SP           ;...ENTRY POINT
        ADD   RØ,@SP
        JSR   PC,@(Sp)+         ;CALL DRIVER
        RTS   R5                ;EXIT TO CALLER
```

The calling program, or a subroutine of the type just illustrated, may also wish to take advantage of a further feature mentioned earlier: the fact that when a driver is already occupied its first word must be

non-zero. The driver itself does not clear this word except in special cases shown in the description for the driver concerned. If the program itself always ensures that it is set to zero between driver tasks, this word forms a suitable Driver-busy flag. Under DOS, in fact, the program parameter list is extended to allow additional words to provide linkage between lists as a queue of which the list indicated in the driver first word is the first link.

The preceding paragraphs are intended merely to indicate possible ways of incorporating the drivers available into the type of environment for which they were designed. The user will probably find others. However, he should read carefully the more detailed description of the driver structure in Appendix A and the individual driver specifications before determining the final form of his program.

In particular, one general word of warning is appropriate here. Although most drivers normally set up an operation and then wait for an interrupt to produce a completion state, there are some cases in which the driver can finish its required task without an interrupt, e.g., "opening" a paper-tape reader involves only a check on its status. Moreover, where "Special Functions" are concerned, the driver routine may determine from the code indicated that the function is not applicable in its case and will, therefore, have nothing to do. In those cases, the driver clears the intermediate return address from the processor stack and takes the completion return immediately. Special problems may arise, however, if the driver concerned may be covering several tasks, any of which may cause a queue for the driver's services under DOS. To overcome these problems, the driver expects to be able to refer to flags outside the scope of the list described so far. This may mean that a program using such a driver may also need to extend the list range to cover this possibility. Extreme care will then be needed.

APPENDIX A

I-O DRIVERS WITHIN THE DISK OPERATING SYSTEM

The principal function of an I/O driver is to satisfy the requirement of a Monitor processing routine for the transfer of a block of data in a standard format to or from the device it represents. This will involve both setting up the device hardware registers to cause the transfer and its control under the interrupt scheme of PDP-11, making due allowance for peculiar device characteristics (e.g., conversion to or from ASCII if some special code is used).

It may also include routines for handling device start-up or shut-down such as punching leader or trailer, and for making available to the user certain special features of the device, such as rewind of magtape.

A.1 Driver Structure

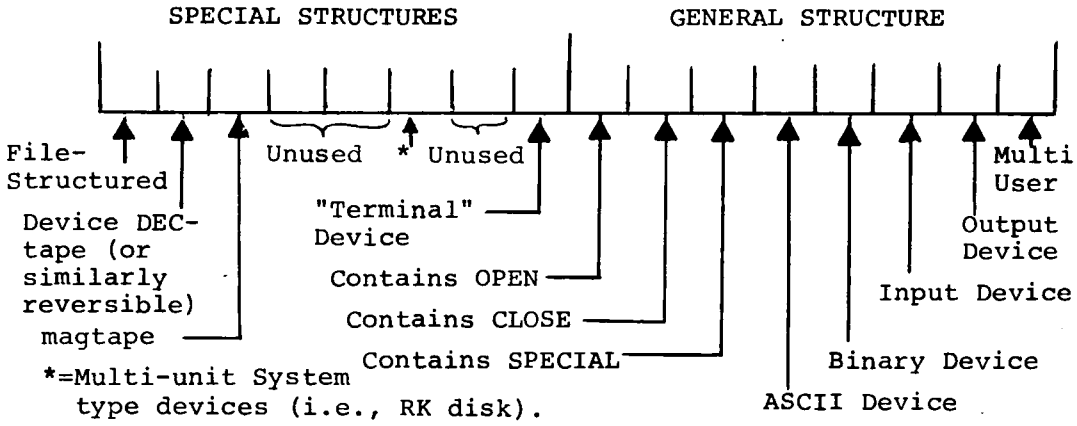
In order to provide a common interface to the monitor, all drivers must begin with a table of identifying information as follows:

DVR:

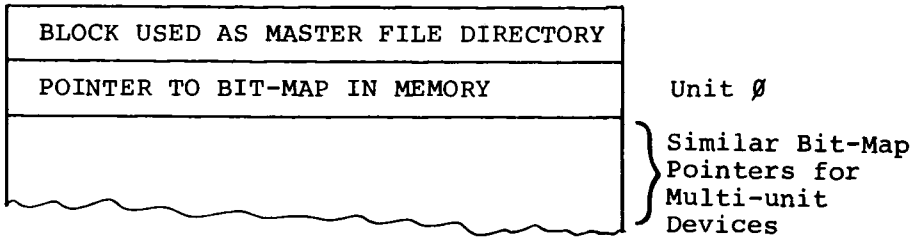
BUSY FLAG (initially Ø)	
FACILITY INDICATOR (expanded below)	
Offset to Interrupt Routine*	Standard Buffer Size in 16-word Units.
Offset to OPEN Routine *	Priority for Interrupt Service
Offset to CLOSE Routine *	Offset to Transfer Routine *
Space	Offset to Special Functions*
DEV	NAME (Packed Radix-5Ø)

Offsets marked * will enable calling routine to indicate routine required. They will be considered as an unsigned value to be added to the start address of the driver. This may mean that with a 256 maximum, the instruction referenced by the offset will be JMP or BR (routine).

Bits in the Facility Indicator Word define the device for monitor reference:



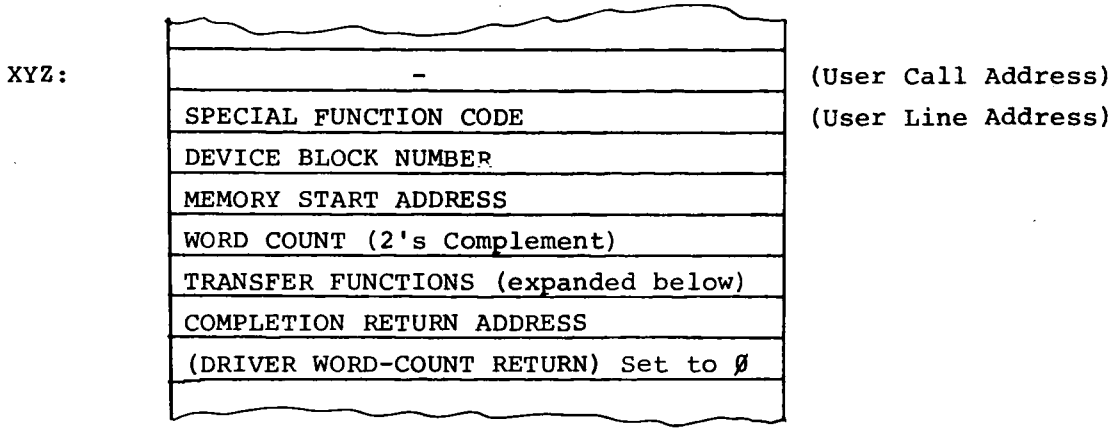
The table should be extended as follows if the device is file-structured:



The driver routines to set up the transfer and control it under interrupt, and possibly for OPEN, CLOSE, and SPECIAL, follow the table. Their detailed operation will be described later.

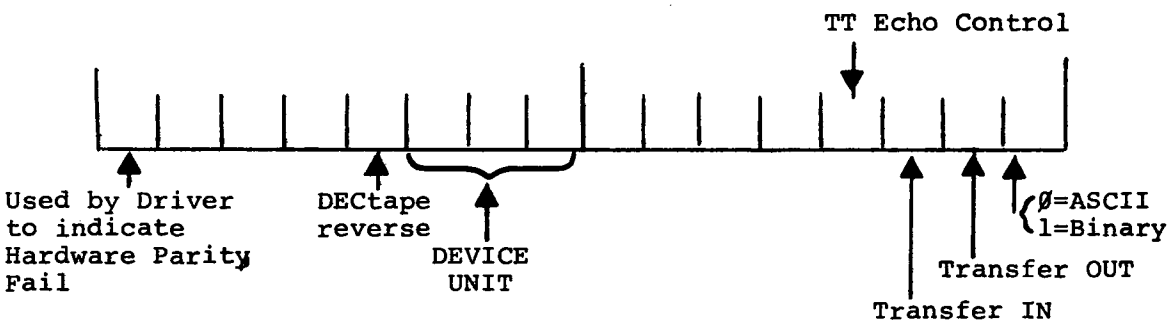
A.2 Monitor Calling

When a Monitor I/O processing routine needs to call the driver, it first sets up the parameters for the driver operation in relevant words of the appropriate DDB[1], as follows:



[1]Dataset Data Block - in full, a 16-word table which provides the main source of communication between the Monitor drivers and a particular set of data being processed on behalf of a using program.

The relevant content of the Transfer Function word is as follows:



Provided that the Facility Indicator in the Driver Table described above shows that the driver is capable of satisfying the request, both from the point of view of direction and mode and of the service required, the Monitor routine places in Register 1 the relative byte address of the entry in the Driver Table containing the offset to the routine to be used (e.g., for the Transfer routine, this would be 1Ø). It then calls the Driver Queue Manager, using JSR PC,S.CDB.

The Driver Queue Manager ensures that the driver is free to accept the request, by reference to the Busy Flag (Word Ø of the driver table). If this contains Ø, the Queue Manager inserts the address of the DDB from Register Ø and jumps to the start of the routine in the driver using Register 1 content to evaluate the address required. If the driver is already occupied, the new request is placed in a queue linking the appropriate DDB's for datasets waiting for the driver's services. It is taken from the queue when the driver completes its current task. (This is done by a recall to the Queue Manager from the routine just serviced, using JSR PC,S.CDQ.)

On entry to the Driver Routine, therefore, the address following the Monitor routine call remains as the "top" element of the processor stack. It can be used by the driver in order to make an immediate return to the Monitor (having initiated the function requested), using RTS PC. It should also be noted that the Monitor routine will have saved register contents if it needs them after the device action. The driver may thus freely use the registers for its own operations.

When the driver has completely satisfied the Monitor request, it should return control to the Monitor using the address set into the DDB. On such return, Register Ø must be set to contain the address of the DDB just serviced and since the return will normally follow hardware interrupt, Registers Ø-5 at the interrupt must be stored on top of the stack.

A.3 Driver Routines

A.3.1 TRANSFER

The sole purpose of the TRANSFER routine is to set the device in motion. As indicated above, the information needed to load the hardware registers is available in the DDB, whose address is contained in the first word of the driver. Conversion of the stored values is, of course, the function of the routine. It must also enable the interrupt; however, it need not take any action to set the interrupt vectors as these will have been preset by the Monitor when the driver is brought into core. Having then given the device GO, an immediate return to the calling processor should be made by RTS PC.

A.3.2 Interrupt Servicing

The form of this routine depends upon the nature of the device. In most drivers it will fall into two parts, one for handling the termination of a normal transfer and the other to deal with reported error conditions.

For devices which are word or byte-oriented, the routine must provide for individual word or byte transfers, with appropriate treatment of certain characters (e.g., TAB or Null) and for their conversion between ASCII or binary and any special device coding scheme, until either the word count in the DDB is satisfied or an error prevents this. On these devices, the most likely cause for such error is the detection of the end of the physical medium; its treatment will vary according to whether the device is providing input or accepting output. The calling program will usually need to take action in the former case and the driver should merely indicate the error by returning the unexpired portion of the word count in DDB Word 7 on exit to the Monitor. Output End of Data, however, will, in general, require operator action. To obtain this, the driver should call the Error Diagnostic Print routine within the Monitor by:

```
MOV    DEVNAM,-(SP)        ;SHOW DEVICE NAME
MOV    #4Ø2,0(SP)          ;SHOW DEVICE NOT READY
IOT                        ;CALL E.D.P.
```

On the assumption that the operator will reset the device for further output and request continuation, the driver must follow the above sequence with a Branch or Jump to produce the desired resumption of the transfer.

Normal transfer handling on blocked devices (or those like RF11 Disk which are treated as such) is probably simpler since the hardware takes care of individual words or bytes and the interrupt only occurs on completion. Errors may arise from many more causes, and their handling is, as a result, much more complex and device dependent. In general, those which indicate definite hardware malfunctions must lead to the situation in which the operator must be informed by diagnostic message and the only recourse after rectification will be to start the program over.

At the other end of the scale there are errors which the driver itself can attempt to overcome by restarting the transfer - device parity failure on input is a common example. If a retrial, or several, still does not enable a satisfactory conclusion, the driver should normally allow programmed recovery and merely indicate the error by Bit 17 of DDB word 5. Nevertheless, because the program may wish to process the data despite the error, the driver should attempt to transfer the whole block requested if this has not already been effected. Between these two extremes, the remaining forms of error must be processed according to the type of recovery deemed desirable.

Whether the routine uses processor registers for its operation or not will naturally depend on considerations of the core space saved against the time taken to save the user's content. However, on completion (or error return) to the Monitor, as indicated in an earlier paragraph, the calling routine expects the top of the stack to contain the contents of all Registers Ø-5 and Register Ø to be set to the address of the DDB just serviced. The driver must, therefore, provide for this.

A.3.3 OPEN

This routine need be provided only for those devices for which some hardware initialization is required by the user. It should not

normally appear in drivers for devices used in a file-oriented manner. Its presence must be indicated by the appropriate bit (Bit 7) in the driver table Facility Indicator.

The routine itself may vary according to the transfer direction of the device. For output devices, the probable action required is the transmission of appropriate data, e.g., CR/LF at a keyboard terminal, form-feed at a printer, or null characters as punched leader code, and for this a return interrupt is expected. The OPEN routine should then be somewhat similar to that for TRANSFER in that it merely sets the device going and makes an interim return via RTS PC, waiting until completion of the whole transmission before taking the final return address in the DDB.

On the other hand, an input OPEN will likely consist of just a check on the readiness of the device to provide data when requested. In this case, the desired function can be effected without any interrupt wait. The routine should, therefore, take the completion return immediately. Nevertheless, it must ensure that the saved PC value on top of the stack from the call to S.CDB is appropriately removed before exit. In the case of drivers which can only service one dataset at a time (i.e., Bit Ø of their Facility Pattern word is set to Ø) and can never, therefore, be queued, it will be sufficient merely to use TST (SP)+ to effect this. A multi-user driver, however, must allow for the possibility that it may be recalled to perform some new task already waiting in a queue. This is shown by the byte at DDB-3 being non-Ø. In this case, the intermediate return to the routine originally requesting the new task has already been made directly by S.CDB. The address now on top of the stack is the return to the routine, whose task the driver has just completed and which has called S.CDQ to dequeue the driver. This return must be taken when the first routine has performed its Completion Return processing. Moreover, this first routine expects to exit as from an interrupt. When a driver is recalled from a queue, it must simulate this interrupt. A possible sequence might be:

```
        MOV   DRIVER, RØ        ;PICK UP DDB ADDRESS
        MOV   (SP)+,R5          ;SAVE INTERIM RETURN
        TSTB  -3(RØ)            ;COME FROM QUEUE?
        BEQ   EXIT
        MOV   @#177776,-(SP)    ;IF SO, STORE STATUS
        MOV   R5,-(SP)          ;...& RETURN
        SUB   #14,SP            ;DUMMY SAVE REGS
EXIT:   JMP   @14(RØ)
```

A.3.4 CLOSE

As with OPEN, this routine should provide for the possibility of some form of hardware shut down such as the punching of trailer code and is not necessary for file-structured devices. Moreover, it is likely to be a requirement for output devices only. If it is provided, Driver Table Facility Indicator (Bit 6) must be set.

Again, the probable form is initialization of the hardware action required, with immediate return via RTS PC and eventual completion return via the DDB-stored address.

A.3.5 SPECIAL

This routine may be included if either the device itself contains the hardware to perform some special function or there is a need for software simulation of such hardware on other devices, e.g., tape rewind. It should not be provided otherwise. Its presence must be indicated by Bit 5 of the Facility Indicator.

The function itself is stored by the Monitor as a code in the DDB as shown earlier. When called, the driver routine must determine whether such function is appropriate in its case. If not, the completion return should be taken immediately with prior stack clearance, as discussed under OPEN. For a recognized function, the necessary routine must be provided. Again, its exit method will depend upon the necessity for an interrupt wait or otherwise.

A.4 Drivers for Terminals

The rate of input from terminal devices is normally dictated externally by the operator, rather than being program-driven; moreover, for both input and output, the amount of data to be transferred on each occasion may be a varying value, i.e., a line rather than a block of standard size. Furthermore, there may be problems with the conflict between echo of input during output. As a result, drivers for such devices will demand special treatment.

Normal output operation, i.e., .WRITE by the program, is handled by the Monitor Processor. On recognizing that the device being used is a terminal, as shown by Bit 8 of the facility indicator, this routine always causes a driver transfer at the end of the user line, even though the internal buffer has not been filled. The driver, however, is given the whole of a standard buffer, padded as necessary with

nulls. Provided the driver can ignore these, the effect is that of just a line of output.

Input control on the other hand, must remain driver responsibility. Overcoming the rate problem will, in most cases, require circular buffering within the driver until demanded by the Monitor. At this point, transfer of data already in should occur. If this is sufficient to fill the monitor buffer, the driver can await the next request before further transfer onward. If insufficient, it should operate as any other device and use subsequent interrupts to continue to satisfy the Monitor request. It must, nevertheless, stop any transfer at the end of a line in normal operation. In order to allow the Monitor to continue, the driver must simulate the filling of the buffer by null padding (of no consequence, since terminals are by nature character-based). (Normal operation, of course, means response to user .READ's and is indicated by the size of the buffer to be filled, namely the driver standard. Should the user be requesting .TRAN'S, the buffer size will vary from the standard in all likelihood and the driver may then assume he requires operation as a normal device -- complete buffer fill-up before return.)

Where input echo is a further complexity, there will doubtless be other requirements. If the echo is made immediately after the input, it may be desirable to have a second buffer to cater for the likely situation that the echo will not exactly match its origin. On the other hand, if the echo is held for any length of time, perhaps to provide correct relations between program-driven output and the echo, the second buffer could be too expensive. A larger input buffer and routines to allow for several outputs to one input character while sitting on that character might be more convenient. The conflict between such echo and program-driven output will require controlled switching within the driver input and output handlers.

HOW TO OBTAIN SOFTWARE INFORMATION

Announcements for new and revised software, as well as programming notes, software problems, and documentation corrections are published by Software Information Service in the following newsletters.

Digital Software News for the PDP-8 & PDP-12
Digital Software News for the PDP-11
Digital Software News for the PDP-9/15 Family

These newsletters contain information applicable to software available from Digital's Program Library, Articles in Digital Software News update the cumulative Software Performance Summary which is contained in each basic kit of system software for new computers. To assure that the monthly Digital Software News is sent to the appropriate software contact at your installation, please check with the Software Specialist or Sales Engineer at your nearest Digital office.

Questions or problems concerning Digital's Software should be reported to the Software Specialist. In cases where no Software Specialist is available, please send a Software Performance Report form with details of the problem to:

Software Information Service
Digital Equipment Corporation
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

These forms which are provided in the software kit should be fully filled out and accompanied by teletype output as well as listings or tapes of the user program to facilitate a complete investigation. An answer will be sent to the individual and appropriate topics of general interest will be printed in the newsletter.

Orders for new and revised software and manuals, additional Software Performance Report forms, and software price lists should be directed to the nearest Digital Field office or representative. U.S.A. customers may order directly from the Program Library in Maynard. When ordering, include the code number and a brief description of the software requested.

Digital Equipment Computer Users Society (DECUS) maintains a user library and publishes a catalog of programs as well as the DECUSCOPE magazine for its members and non-members who request it. For further information please write to:

DECUS
Digital Equipment Corporation
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

PDP-11 Device Driver Package
DEC-11-NIZB-D
February 1972.

READER'S COMMENTS

Digital Equipment Corporation maintains a continuous effort to improve the quality and usefulness of its publications. To do this effectively we need user feedback -- your critical evaluation of this manual.

Please comment on this manual's completeness, accuracy, organization, usability, and readability.

Did you find errors in this manual? If so, specify by page.

How can this manual be improved?

Other comments?

Please state your position. _____ Date: _____

Name: _____ Organization: _____

Street: _____ Department: _____

City: _____ State: _____ Zip or Country _____

— — — — — — — — — — — — — — — — Fold Here — — — — — — — — — — — — — — — — — —

— — — — — — — — — — — — — Do Not Tear - Fold Here and Staple — — — — — — — — — — —

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

digital

Digital Equipment Corporation
Software Information Services
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

DEC-11-RICB-D

P D P - 1 1

C R 1 1 / C M 1 1 C A R D R E A D E R D R I V E R

JANUARY 1972

SUPPLEMENT TO:

PDP-11 DEVICE DRIVER PACKAGE

DEC-11-NIZA-D

Copyright © 1971, 1972 by Digital Equipment Corporation

THIS DOCUMENT IS FOR INFORMATION PURPOSES
AND IS SUBJECT TO CHANGE WITHOUT NOTICE.

pdp11 PROGRAM LIBRARY

CONTENTS

Section			Page
1.0	BASIC DRIVER (ASCII ONLY) - CR11		1
	1.1	Driver Table	1
	1.2	Service Routines	2
	1.2.1	OPEN	2
	1.2.2	TRANSFER	2
	1.2.3	Interrupt Service	2
	1.2.4	Error Handling	4
	1.3	Alternative Drivers for ASCII Only Usage	4
	1.3.1	DEFAULT	4
	1.3.2	ONLY26/ONLY29	4
	1.3.3	BLANKS	5
2.0	READING OF BINARY CARDS		5
	2.1	BINARY FORMAT	6
	2.2	CODING CHANGES FOR BINARY OPERATIONS	7
	2.2.1	Driver Table	7
	2.2.2	OPEN	7
	2.3	TRANSFER	8
	2.4	INTERRUPT SERVICING	8
	2.5	ERROR HANDLING	9
3.0	CM11 MARK SENSE READER DRIVER		9
4.0	DETAILED IMPLEMENTATION		9
APPENDIX			
A	PDP-11 Card Codes		A-1
B	NOTES ON ALGORITHMS USED IN CR11/CM11 CARD READER DRIVER		B-1
	B.1	Hollerith to ASCII Conversion	B-1
	B.2	Binary Packing	B-1
	B.3	Switching	B-2
C	UNPACKING BINARY DATA FROM THE CR11/CM11 CARD READER DRIVER -- A SUGGESTED ALGORITHM		C-1
D	PREPARATION AND USAGE OF CR11/CM11 CARD READER DRIVER		D-1
	D.1	Preparation	D-1
	D.1.1	Assembly	D-1
	D.1.2	Linking	D-2
	D.1.3	Inclusion in DOS Monitor Library	D-2
	D.2	Usage	D-2

CR11/CM11 CARD READER DRIVER

The card-reader driver performs device-dependent I/O functions for the PDP-11 CR11 Card Reader Control within the Disk Operating System (DOS). At each Monitor request on behalf of a running program, the driver, in its basic version, reads a single card, which may be punched in either 026 or 029 Hollerith notation as indicated by specially coded cards in the input deck. The resultant data is stored in a specified area as a line of up to 80 ASCII characters terminated by a carriage-return/line-feed.

By conditional assembly of its source, however, the driver may be produced in various versions to include the following additional features:

- Restriction of input conversions to one type of punch.
- Automatic deletion of card-columns 73-80 and of trailing spaces from preceding columns.
- Reading of cards punched in a binary format with data passed to the user, packed 4 columns to 3 words.
- Provision of similar facilities for the 40-column Mark Sense Reader under CM11 control.

All cards are read under the PDP-11 interrupt system. The driver, therefore, contains the routines needed, firstly, to initiate a card transfer and, secondly, to service the interrupt as each column is read and supply the required conversion of its content until the end of the card is seen. An OPEN function is also included to enable a using program to ensure that the reader is on-line before issuing its first read. CLOSE and Special Functions processing is unnecessary and is not provided.

1.0 BASIC DRIVER (ASCII ONLY) - CR11

The driver is in two parts: the Driver Table and the Service Routines.

1.1 Driver Table

The table occupies the first seven words of the driver in the standard format for I/O drivers under DOS. It includes the following particular information:

- Capabilities: Single user
  Input in ASCII only
  Non-file structured

- Standard buffer size = 96 bytes
- Interrupt servicing at priority level 6
- Device Name: CR

1.2 SERVICE ROUTINES

1.2.1 OPEN

The OPEN routine first checks the Control Status of the reader. If for any reason this is off-line, printing of an AØØ2 error message (device not ready) is requested. If a return is made, the check and message are repeated until an on-line state is detected. The routine then prepares the driver to accept 029 punching by default and returns control to the calling Monitor routine.

1.2.2 TRANSFER

Using the starting address set into its first word by the calling routine, the driver's TRANSFER processor accesses the DDB for the dataset requiring the card input to extract and store internally pointers to the start and end of the buffer area for the data. The first word of the buffer is then cleared as an indicator that the first column is yet to be read. The routine returns to the Monitor with the Reader Control set to INT ENB and GO.

1.2.3 Interrupt Service

At each interrupt, a check is first made for error or card-done conditions. If neither is seen, the column data just accessed is used to compute an index into a table of associated parity-ASCII characters (see Appendix A), the relevant character is extracted and stored as a byte in the buffer provided. The next buffer byte is set to a positive non-zero value to show that a valid read is under way. An interrupt return is then taken.

For card column 1, however, checks are also made for a card with any of the following special codings in that colum:

- 12-11-8-9 — This indicates that the cards which follow are to be read as 026 punch codes, and on recognition of this an internally stored offset is modified to use the appropriate section of the table of ASCII values.
- 12-0-7-9 — This indicates that the following cards contain 029 and cause similar offset modification.

- 12-11-0-1 This indicates the end of the card file, and a card so coded must be present ("Hopper Empty" is merely deemed a "Device not ready" state to allow usage of very large decks). When this card is seen, the next buffer byte is set negative to show EOF. Since no data will now be forthcoming, the appropriate word is set in the dataset DDB to show this.

When any of the three cases is seen, the Reader Control Status is reset to EJECT before the interrupt exit is taken, thereby causing the remainder of the card to be ignored.

The rest of a card is similarly ejected if, during the processing of any column, the buffer is filled. In normal READ operations for which the Monitor provides a standard-sized buffer of 96 bytes, this cannot occur. This is not necessarily the case if the user program has requested TRAN. If this program also supplies short buffers, this can mean the possible loss of card data, intentionally or otherwise.

When a card-done condition is detected, the Reader interrupt is disabled. The underway state, shown as noted earlier in the next buffer byte, is then checked. If no data has yet been processed because the card just read merely contained a control code, a new card transfer is requested by recalling the TRANSFER routine. Otherwise, the unused portion of the buffer provided is cleared and the parity-ASCII values for RETURN and LINE FEED are inserted to follow the last data read (in the short buffer, these will overwrite the last two columns processed). As required by the general driver specification, the service routine then saves all user registers on the processor stack and takes the supplied completion return with Register 0 set to the address of the DDB just serviced.

It should be noted that this process allows the reading of only one card at each request, regardless of the size of the buffer provided. Because a card-read (once it has begun) must continue to completion, any attempt to fill the unused buffer space must necessitate the internal storage of any overflow, if possible loss of data is to be avoided. In keeping the size of the driver to a reasonable limit, the provision of such internal storage is not considered desirable. For the READ form of I/O, the buffer supplied by the Monitor must be

excessive, as space is allocated in 16-word units; the null padding, however, is not passed on to the user program. On the other hand, it can be seen that no advantage is gained by a program defining a buffer larger than 82 bytes when using the device-dependent TRAN.

1.2.4 Error Handling

The detection of any error condition is taken to mean a "Device not ready" state, leading to the printing of an AØØ2 message with the reader interrupt disabled. If the operator requests resumption by a CONTINUE command at the keyboard, the error processor will recall the TRANSFER routine to repeat the read and exit to await a fresh interrupt. This allows the operator to rectify the error, if possible: the card causing the error should be replaced as the first to be read after the resumption.

NOTE

A "Hopper Empty" condition is detected before the last card has been processed. It is, therefore, essential that the EOF card for a deck be followed by at least one more card (can be blank). Should this be omitted, normal completion can be effected by re-insertion of the EOF card followed by a blank card.

1.3 ALTERNATIVE DRIVERS FOR ASCII ONLY USAGE

As has been shown in the previous section, the standard driver accepts cards punched in either 026 or 029 codes when so directed by control cards, or the driver assumes 029 by default. Unless the user program then requests input by TRAN with short buffer sizes, 80-character lines are the norm. To provide other versions of the driver more suited to the needs of a particular installation, the following conditional assembly parameters have been included in the source language. If these are defined when the source is processed (DEFALT = Ø is sufficient definition), the driver will operate as indicated.

1.3.1 DEFAULT

This forces the driver to assume 026 card codes as the default. Control cards as defined, however, will still override this assumption. The effect on the driver length is negligible - one word.

1.3.2 ONLY 26/ONLY 29

If the user has only one type of punch, he can restrict the driver accordingly by the definition of the relevant one of these parameters.

In this case, control cards will have no effect and will be ignored if present. Because the driver then needs only half of its conversion tables and certain checks can be eliminated, the driver size is reduced by some 45 words.

1.3.3 BLANKS

By common practice, card columns 73-80 are often used only for control information, e.g., sequence numbering, which need not be processed by the using program. Moreover, quite a number of columns before these may contain nothing but blanks (translated into spaces in ASCII). Although cards of this type will be accepted by systems programs such as Assembler or Editor without error, the burden of lines always 80 characters long can be excessive, especially if, as one example, the only means for listing the assembly of a card source is a teleprinter.

The parameter BLANKS has been included to enable the user optionally to remove this burden, provided that he is also prepared to accept an increase of some 18 words in driver size. The driver in this case will still continue to transfer 80-characters as its normal operation. If, however, the card deck is preceded by a control card punched 12-11-0-7-8-9 in column 1, or at any point contains a card so punched, columns 73-80 in all subsequent cards will be ignored and the CR/LF terminating the line each card represents will be set immediately after the last non-blank data column. The automatic deletion will remain until the user program requests an OPEN for a fresh deck.

NOTE

DEFALT, ONLY26, and ONLY29 are of course mutually exclusive. BLANKS, on the other hand, may be defined alone or with any one of the other three.

2.0 READING OF BINARY CARDS

Some users may wish to have the additional facility of reading cards directly as 12 bits per column rather than as ASCII characters, perhaps for one of the following situations:

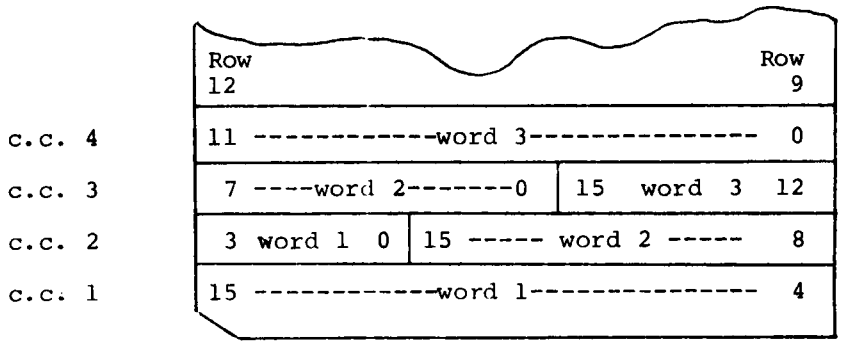
- Linking or loading of card programs produced by cross-assemblers or linkers developed on other computers.
- Processing of binary data output by other computers.

- Reading of cards using character codes other than those associated with 026 or 029 punches[1].

A further conditional assembly parameter, BINARY, has been included in the driver source to meet this requirement.

2.1 BINARY FORMAT

The driver, assembled with this parameter defined, still continues to function exactly as described earlier whenever the using program requests ASCII input. If, however, a binary transfer is called, the processing Monitor routine will inform the driver of this by setting to 1 bit Ø of the status word of the DDB for the dataset concerned (DDB+12). On recognition of this, the driver accepts each column as 12 data bits and passes it to the program in a packed form, four columns taking three words, in accordance with the following pattern:



This format, which is compatible with that used by IBM 1130 and 1800 Series, has been chosen because it alone provides for all the facilities listed above and, moreover, is compatible with the device-independent philosophy of DOS. It may nevertheless mean that the user who needs to process each column on a word basis must include in his program the routine to unpack again the data from the driver (a possible algorithm is offered in Appendix C).

The main effect of the inclusion of the binary capability in the driver is a substantial increase in its size, hence the reason for this not being made a feature of the standard version. Apart from the coding changes need to cater for the different processing (which are outlined in subsequent paragraphs), the assumption made in the case of ASCII data, that null padding in oversize buffers can be safely overlooked, no longer applies. Each buffer word must always be considered

[1]An alternative in this case might be to change the driver's conversion tables to satisfy the different codes, provided that these use the same pattern - null, 0, 11 or 12 in association with one punch in range 1-7, perhaps with a punch in 8 or 9.

as valid data. In order that the driver may cope with this situation (the Monitor can only supply a buffer made up of 16-word units), it must now contain its own internal buffer to hold any residue from a card used to fill the remaining Monitor area. Allowing that such residue can be stored already packed in its final form, the internal buffer must be at least 60 words long. Together with the additional coding, the driver increases from its 200 words, in the basic ASCII-only form, to 380. Against this, however, the presence of the internal buffer also means that the driver can supply valid binary data into user buffers of any length for a program issuing TRAN requests.

It should be noted, too, that the format used does not in itself provide any checking upon the read accuracy of each card. All 80 columns are assumed to contain actual data. Programs READing in unformatted binary modes or using TRAN must make their own checks if these are important, in just the same way as with other drivers. On the other hand, the Monitor processing formatted binary READs will expect the data to conform to its normal standards for each request.

```
Byte Ø:     =1
Byte 1:     =Ø
Bytes 2-3:  Number of bytes to be read including bytes
            Ø-3 but not the final checksum.
Bytes 4-N:  DATA BYTES
Byte N+1:   Checksum of Bytes Ø-N
```

If program developed to produce binary cards in such format also punch one card for each READ, the data checksum can serve as card checksum as well (in this case, nulls following the checksum will be ignored).

2.2 CODING CHANGES FOR BINARY OPERATIONS

The changes in the driver's operations brought about by the definition of BINARY are as follows:

2.2.1 DRIVER Table

Capabilities as indicated in Section 1.1 are extended to include binary input.

2.2.2 OPEN

The OPEN processor still first checks the on-line state of the reader and takes appropriate action as described earlier if it is not ready. It now, however, anticipates the fact that after its exit, the driver will be recalled immediately to fill the Monitor buffer against the program's first READ. At this time the Monitor will be unable to di-

rect the driver on the mode of reading. The OPEN routine therefore sets a switch to cause a return to be made without a card transfer, when this recall occurs. It also means that the Monitor will give the user program 96 bytes of null (equivalent to leader on a paper tape) for its first unformatted binary operations. (Incidentally, the switch is set to perform a proper read when the driver is loaded into memory; if, therefore, the program does not request an OPEN but starts by a READ, the correct result will occur.)

2.3 TRANSFER

As with all drivers, the card reader driver must contain only position-independent code. To control its internal buffer, however, it needs absolute pointers. The first actual read causes execution of some once-only code to establish these. Again, a switch effects this. A further switch is then set according to the mode in which the data is to be handled. As mentioned earlier, if this is ASCII, the code for the standard version of the driver is followed, both during the TRANSFER and INTERRUPT service functions.

For binary transfers, any data remaining from a previous read is passed to the Monitor buffer immediately. If this is sufficient to satisfy the Monitor's requirements, an immediately completion return is taken. (Since this would normally follow an interrupt and the Monitor will expect this, the driver must in this case simulate the appropriate conditions, i.e., leave an interrupt exit on the stack, supposedly preceded by saved registers.) A new card read is initiated in the same way as ASCII if more data is needed. In addition, the second byte in the Monitor buffer not yet filled is cleared as a switch for use by the packing algorithm which handles odd and even card columns differently (see Appendix B).

2.4 INTERRUPT SERVICING

The packing of binary data is accomplished as each column is read. At the beginning of each card a check is again made for EOF. Unlike the ASCII case, the coding of a single column cannot provide a unique identification. The same pattern (12-11-0-1) is therefore looked for in each of the first eight columns before the end-of-data is signalled and the remainder of the card ejected. (The same card can still be used for either data form; the ASCII processor merely uses the first column punched.) No other control cards are expected in binary mode.

In this case, control cards will have no effect and will be ignored if present. Because the driver then needs only half of its conversion tables and certain checks can be eliminated, the driver size is reduced by some 45 words.

1.3.3 BLANKS

By common practice, card columns 73-80 are often used only for control information, e.g., sequence numbering, which need not be processed by the using program. Moreover, quite a number of columns before these may contain nothing but blanks (translated into spaces in ASCII). Although cards of this type will be accepted by systems programs such as Assembler or Editor without error, the burden of lines always 80 characters long can be excessive, especially if, as one example, the only means for listing the assembly of a card source is a teleprinter.

The parameter BLANKS has been included to enable the user optionally to remove this burden, provided that he is also prepared to accept an increase of some 18 words in driver size. The driver in this case will still continue to transfer 80-characters as its normal operation. If, however, the card deck is preceded by a control card punched 12-11-0-7-8-9 in column 1, or at any point contains a card so punched, columns 73-80 in all subsequent cards will be ignored and the CR/LF terminating the line each card represents will be set immediately after the last non-blank data column. The automatic deletion will remain until the user program requests an OPEN for a fresh deck.

NOTE

DEFALT, ONLY26, and ONLY29 are of course mutually exclusive. BLANKS, on the other hand, may be defined alone or with any one of the other three.

2.0 READING OF BINARY CARDS

Some users may wish to have the additional facility of reading cards directly as 12 bits per column rather than as ASCII characters, perhaps for one of the following situations:

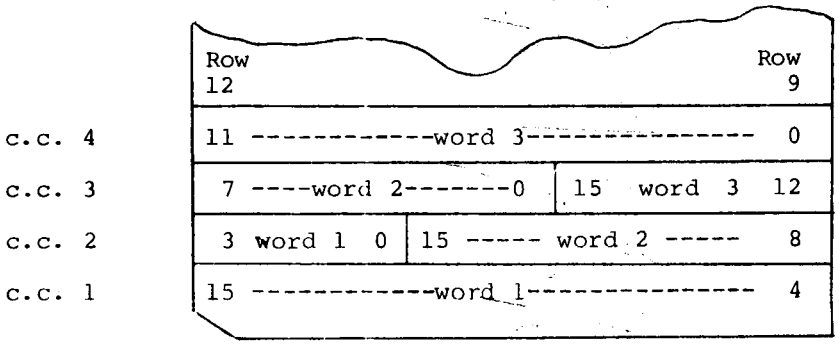
- Linking or loading of card programs produced by cross-assemblers or linkers developed on other computers.
- Processing of binary data output by other computers.

- Reading of cards using character codes other than those associated with 026 or 029 punches[1].

A further conditional assembly parameter, BINARY, has been included in the driver source to meet this requirement.

2.1 BINARY FORMAT

The driver, assembled with this parameter defined, still continues to function exactly as described earlier whenever the using program requests ASCII input. If, however, a binary transfer is called, the processing Monitor routine will inform the driver of this by setting to 1 bit Ø of the status word of the DDB for the dataset concerned (DDB+12). On recognition of this, the driver accepts each column as 12 data bits and passes it to the program in a packed form, four columns taking three words, in accordance with the following pattern:



This format, which is compatible with that used by IBM 1130 and 1800 Series, has been chosen because it alone provides for all the facilities listed above and, moreover, is compatible with the device-independent philosophy of DOS. It may nevertheless mean that the user who needs to process each column on a word basis must include in his program the routine to unpack again the data from the driver (a possible algorithm is offered in Appendix C).

The main effect of the inclusion of the binary capability in the driver is a substantial increase in its size, hence the reason for this not being made a feature of the standard version. Apart from the coding changes need to cater for the different processing (which are outlined in subsequent paragraphs), the assumption made in the case of ASCII data, that null padding in oversize buffers can be safely overlooked, no longer applies. Each buffer word must always be considered

[1]An alternative in this case might be to change the driver's comversion tables to satisfy the different codes, provided that these use the same pattern - null, 0, 11 or 12 in association with one punch in range 1-7, perhaps with a punch in 8 or 9.

as valid data. In order that the driver may cope with this situation (the Monitor can only supply a buffer made up of 16-word units), it must now contain its own internal buffer to hold any residue from a card used to fill the remaining Monitor area. Allowing that such residue can be stored already packed in its final form, the internal buffer must be at least 60 words long. Together with the additional coding, the driver increases from its 200 words, in the basic ASCII-only form, to 380. Against this, however, the presence of the internal buffer also means that the driver can supply valid binary data into user buffers of any length for a program issuing TRAN requests.

It should be noted, too, that the format used does not in itself provide any checking upon the read accuracy of each card. All 80 columns are assumed to contain actual data. Programs READing in unformatted binary modes or using TRAN must make their own checks if these are important, in just the same way as with other drivers. On the other hand, the Monitor processing formatted binary READs will expect the data to conform to its normal standards for each request.

```
Byte Ø:     =1
Byte 1:     =Ø
Bytes 2-3: Number of bytes to be read including bytes
           Ø-3 but not the final checksum.
Bytes 4-N: DATA BYTES
Byte N+1:  Checksum of Bytes Ø-N
```

If program developed to produce binary cards in such format also punch one card for each READ, the data checksum can serve as card checksum as well (in this case, nulls following the checksum will be ignored).

2.2 CODING CHANGES FOR BINARY OPERATIONS

The changes in the driver's operations brought about by the definition of BINARY are as follows:

2.2.1 DRIVER Table

Capabilities as indicated in Section 1.1 are extended to include binary input.

2.2.2 OPEN

The OPEN processor still first checks the on-line state of the reader and takes appropriate action as described earlier if it is not ready. It now, however, anticipates the fact that after its exit, the driver will be recalled immediately to fill the Monitor buffer against the program's first READ. At this time the Monitor will be unable to di-

rect the driver on the mode of reading. The OPEN routine therefore sets a switch to cause a return to be made without a card transfer, when this recall occurs. It also means that the Monitor will give the user program 96 bytes of null (equivalent to leader on a paper tape) for its first unformatted binary operations. (Incidentally, the switch is set to perform a proper read when the driver is loaded into memory; if, therefore, the program does not request an OPEN but starts by a READ, the correct result will occur.)

2.3 TRANSFER

As with all drivers, the card reader driver must contain only position-independent code. To control its internal buffer, however, it needs absolute pointers. The first actual read causes execution of some once-only code to establish these. Again, a switch effects this. A further switch is then set according to the mode in which the data is to be handled. As mentioned earlier, if this is ASCII, the code for the standard version of the driver is followed, both during the TRANSFER and INTERRUPT service functions.

For binary transfers, any data remaining from a previous read is passed to the Monitor buffer immediately. If this is sufficient to satisfy the Monitor's requirements, an immediately completion return is taken. (Since this would normally follow an interrupt and the Monitor will expect this, the driver must in this case simulate the appropriate conditions, i.e., leave an interrupt exit on the stack, supposedly preceded by saved registers.) A new card read is initiated in the same way as ASCII if more data is needed. In addition, the second byte in the Monitor buffer not yet filled is cleared as a switch for use by the packing algorithm which handles odd and even card columns differently (see Appendix B).

2.4 INTERRUPT SERVICING

The packing of binary data is accomplished as each column is read. At the beginning of each card a check is again made for EOF. Unlike the ASCII case, the coding of a single column cannot provide a unique identification. The same pattern (12-11-0-1) is therefore looked for in each of the first eight columns before the end-of-data is signalled and the remainder of the card ejected. (The same card can still be used for either data form; the ASCII processor merely uses the first column punched.) No other control cards are expected in binary mode.

After the entire card has been read as indicated by a card-done condition, the TRANSFER routine is recalled to continue its process of transferring the data into the Monitor buffer. Since an interrupt has no occurred, the return to the Monitor on completion is by normal means.

2.5 ERROR HANDLING

Any error condition is again considered a "device not ready" and is handled accordingly. Because a repeat of the TRANSFER routine as a way of resuming read operations on return would perhaps lose data already passed to the Monitor from a previous card, a failure in binary mode leads only to that part of the TRANSFER operations which causes a new card read.

3.0 CM11 MARK SENSE READER DRIVER

The CM11 Control is expected to use only 40-column cards. (The normal CR11 driver with or without definition of the special assembly parameters will function without change if 80-column cards are used.) To provide the following particular benefits in view of the smaller amount of data available at each card read, one more parameter for conditional assembly has been included - MARKS:

- Standard Buffer size = 64 bytes rather than 96.
- Internal buffer for binary operation is reduced from 60 to 30 words.
- If BLANKS has been defined, automatic deletion of trailing spaces will follow recognition of the relevant control card but not of the last 8 columrs.

4.0 DETAILED IMPLEMENTATION

Comments on the listing which follow illustrate the general form of the driver. Further explanation of some of the more obscure techniques used is given in Appendix B. Other appendices summarize the ASCII/Hollerith equivalences, the procedures for obtaining the various versions of the driver, and the comparative sizes of each.

```
;COPYRIGHT 1971, DIGITAL EQUIPMENT CORP., MAYNARD, MASS.

;VERSION NUMBER:          V005A
;                         000
;
;CARD READER DRIVER (CR)
;
;   A)    FOR ASCII INPUT, AT EACH TRANSFER REQUEST
;         ONE CARD WILL BE READ, UP TO 80 CHARACTERS,
;         FOLLOWED BY CR-LF, WILL BE PASSED TO THE
;         CALLING ROUTINE AS SPECIFIED BY THE WORD
;         COUNT GIVEN. (IF THIS IS > 41, REMAINING
;         BYTES WILL BE CLEARED.
;         ALL ERRORS (INCLUDING 'HOPPER EMPTY' UPON
;         AN 'OPEN' CALL) WILL BE TREATED AS 'DEVICE
;         NOT READY'. USER CAN RESUME OPERATION BY
;         RECTIFICATION OF ERROR OR REFILL OF HOPPER
;         AND ENTRY OF 'CO' COMMAND AT KEYBOARD.
;         THE END OF A FILE WILL BE DETERMINED BY
;         RECOGNITION OF A TERMINAL CONTROL CARD:-
;                 12-11-0-1 PUNCHED IN C.C. 1
;
;                 XXXXXXXXXXXXXXXXXXXXXXXX
;                 X                      X
;                 X        NOTES         X
;                 X                      X
;                 XXXXXXXXXXXXXXXXXXXXXXXX
;
;   1)    THIS DRIVER CAN BE ASSEMBLED FOR USE
;         IN CONNECTION WITH EITHER '026' OR '029'
;         PUNCHES OR BOTH AS INDICATED BY PARAMETER
;         SPECIFICATION AT START OF SOURCE INPUT
;         AS FOLLOWS:-
;
;         A)  "ONLY26=0"  -  READ ONLY '026' CODES.
;         B)  "ONLY29=0"  -  READ ONLY '029' CODES
;         C)  "DEFALT=0"  -  READ BOTH TYPES OF CODE
;                            WITH '026' AS DEFAULT
;         D)     NTL      -  READ BOTH TYPES OF CODE
;                            WITH '029' AS DEFAULT
;
;         IN CASES (C) & (D), DRIVER WILL USE DEFAULT
;         UNLESS DIRECTED OTHERWISE BY ENTRY OF A
;         CONTROL CARD PUNCHED IN C.C. 1:-
;
;                 12-0-7-9  =  '029' CODES FOLLOW
;                 12-11-8-9 =  '026' CODES FOLLOW
;
;   2)    IF PARAMETER "BLANKS" IS DEFINED, C.C. 73-80
;         & TRAILING SPACES BEFORE THESE WILL BE DISCARDED,
;         WITH 'CR-LF' FOLLOWING LAST VALID DATA, PROVIDED
;         THAT CARD FILE IS PRECEDED BY CTL CARD WITH
;         12-11-0-7-8-9 PUNCHED IN C.C. 1. IN THIS CASE
;         HOWEVER, IF THE USER BUFFER IS </= 82 BYTES,
;         ONLY TRAILING SPACE REMOVAL WILL BE EFFECTED.
```

```
;  B)   IF THE PARAMETER "BINARY" IS DEFINED AT ASSEMBLY,
;       XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
;
;       THIS VERSION WILL ALSO ALLOW READING OF CARDS
;       IN BINARY FORMAT. AS EACH CARD IS READ, 12 BITS FROM
;       ALL 80 COLS WILL BE ACCESSED & STORED IN AN INTERNAL
;       BUFFER IN A PACKED FORM, I.E. 4 COLS = 3 WORDS AS
;       FOLLOWS:-
;               C.C.1 > WORD 1, BITS 15-4
;               C.C.2 > WORD 1, BITS 3-0; WORD 2, BITS 15-8
;               C.C.3 > WORD 2, BITS 7-0; WORD 3, BITS 15-12
;               C.C.4 > WORD 3, BITS 11-0
;
;       THE PACKED FORM WILL BE TRANSFERRED TO THE USER
;       BUFFER UNTIL THIS IS FILLED, ANY DATA THEN REMAINING
;       IN THE INTERNAL BUFFER BEING RETAINED UNTIL THE
;       NEXT READ REQUEST.
;
;       TREATMENT OF ASCII READING WILL STILL FOLLOW PATTERN
;       DESCRIBED IN THE PREVIOUS PARAGRAPH WITHOUT EXCEPTION.
;
;       THE ONLY CONTROL CARD WHICH WILL HAVE ANY EFFECT
;       IN BINARY READING WILL BE THAT INDICATING E.O.F. (IN
;       THIS CASE 12-11-0-1 PUNCHING MUST APPEAR IN AT LEAST
;       C.C. 1 THRU 8).
;
;       N.B. WHEN ASSEMBLED FOR USAGE IN BOTH MODES, AN 'OPEN'
;       CALL WILL NOT CAUSE READING OF A CARD TO ALLOW THIS
;       TO BE TRANSLATED AS REQUIRED BY THE READ MODE
;       SPECIFIED BY THE USER.
;
;  C)   DRIVER CAN ALSO BE USED FOR 82-COLUMN MARK
;       SENSE READER, FOR 44-COLUMN READER, ECONOMIES
;       IN BUFFER SIZE CAN BE OBTAINED BY DEFINITION
;       AT ASSEMBLY OF PARAMETER "MARKS". THIS WILL
;       ALSO PREVENT AUTOMATIC REMOVAL OF COLS 33-44
;       IN 'BLANKS-SUPPRESS' MODE OF USAGE.
;
;               XXXXXXXXXXXXXXXXXXXXXXXXX
;               X                        X
;               X          NOTE          X
;               X                        X
;               XXXXXXXXXXXXXXXXXXXXXXXXX
;
;       PARAMETER DEFINITIONS CAN BE MADE DURING PASS 1
;       OF THE ASSEMBLY ONLY IF READ AS DESCRIBED IN
;       PAL-11R MANUAL, SECTION 9-2, P.6.
;
;               @CR,LP:,/PA:2<KB:/PA:1,DF:CR
;       FOLLOWED BY:-
;               ONLY29=0
;               BINARY=0
;               BLANKS=0
;               ^C
;               .END<CR><LF>
;
;               XXXXXXXXXXXXXXXXXXXXXXXXX
;
```

```
                        .TITLE CR
                        .GLOBL CR
                        ;
000000                  R0=%0
000001                  R1=%1
000002                  R2=%2
000003                  R3=%3
000004                  R4=%4
000005                  R5=%5
000006                  SP=%6
000007                  PC=%7
                        ;INTERFACE TABLE:
000000 000000 CR:       .WORD 0                 ;CURRENT DDB OR 0 IF IDLE
                        .IFDF   BINARY
000002     224          .BYTE   224,0           ;FACILITIES: ASCII INPUT,OPEN IN
000003     000
                        .ENDC
                        .IFDF   BINARY
                        .BYTE   234,0           ;ALLOW BINARY IF READ.
                        .ENDC
                        .IFDF   MARKS
000004     003          .BYTE   3               ;STD BUFFER SIZE = 96 BYTES
                        .ENDC
                        .IFDF   MARKS
                        .BYTE   2               ;(64 IF 40-COL MARK SENSE)
                        .ENDC
000005     104          .BYTE   CR.INT-CR,340   ;INTERRUPT SVCE AT PRL 6
000006     340
000007     022          .BYTE   CR.OPN-CR       ;OFFSET TO OPEN
000010     042          .BYTE   CR.TFR-CR       ;OFFSET TO TRANSFER
000011     000          .BYTE   0,0,0           ;(NO CLOSE OR SPEC. FUNC.)
000012     000
000013     000
000014 012620 CR.NAM:   .RAD50  'CR'
                        ;OPEN PROCESSOR:
000016 004767 CR.OPN:   JSR     PC,CR.NPY
       000356
000022 032737 CR.OPN:   BIT     #400,@#CR.CSR   ;CARDS IN HOPPER?
       000400
       177160
000030 001372           BNE     CR.OPN          ;IF NOT TELL USER TO READY
                        .IFDF   ONLY26&ONLY29
                        .IFDF   DEFALT          ;FOR DUAL-PUNCH DRIVER ...
000032 005027           CLR     (PC)+           ;... SET CONV. TABLE OFFSET
                        .ENDC                   ;... FOR DEFAULT PUNCH
                        .IFDF   DEFALT          ;... AS APPROPRIATE
                        MOV     #104,(PC)+
                        .ENDC
000034 000000 CR.TOS:   .WORD   0
                        .ENDC
                        .IFDF   BLANKS          ;IF BLANK-SUPPRESS VERSION ...
                        CLRB    CR.ZSW          ;... FORCE SUPPRESS OFF
                        .ENDC
                        .IFDF   BINARY
000036 005726           TST     (SP)+           ;IGNORE INTERIM RETURN
000040 000523           BR      CR.EXT          ;... & TAKE COMPLETION
                        .ENDC
                        .IFDF   BINARY          ;FOR BINARY VERSION ...
                        CLRB    CR.TFR          ;... FORCE NO OPEN READ
                        BR      CR.OPN          ;... BY MAKING COMP HERE
                CR.EXT: INCB    CR.TFR          ;... BEFORE EXIT
                        BR      CR.OPN
                        .ENDC
```

```
                        ;SUBSIDIARY ROUTINES:
                        ;  A)  RESTART AFTER ERROR:
                        CR.AGN: .IFDF   BINARY          ;IN BINARY VERSION ...
                                TSTB    CR.ISW          ;... CHECK IF BINARY READ
                                BEQ     CR.TER          ;IF NOT CAN JUST START OVER
                                BR      CR.ERD          ;ELSE LEAVE USER BUFFER ALONE
                        ;  B)  INITIALISE INTERNAL BUFFER POINTERS:
                        CR.ISP: MOV     PC,-(SP)        ;GET BUFFER START
                                ADD     #CR.BUF-.,@SP
                                MOV     @SP,(PC)+
                        CR.IBS: .WORD   0
                                ADD     #CR.BSZ,@SP     ;NOW GET END
                                MOV     @SP,(PC)+       ;STORE AS CONTROL
                        CR.IBF: .WORD   0
                                MOV     (SP)+,(PC)+     ;... & AS INIT. PTR
                        CR.IBP: .WORD   0
                                INCB    @PC             ;MUSTNIT COME HERE AGAIN!
                                .ENDC

                        ;TRANSFER SET-UP PROCESSOR:
                        CR.TER: .IFDF   BINARY          ;FOR BINARY VERSION ...
                                BR      .+4             ;... SWITCH TABLE FORCES ...
                                BR      CR.NXT          ;... CORRECT INIT.
                                BR      CR.ISP
                                .ENDC
000042 016700           MOV     CR,R0           ;GET DDB ADDRESS
       177732
000046 062700           ADD     #6,R0           ;... & MOVE TO BUFFER STORE
       000006
000052 012046           MOV     (R0)+,-(SP)     ;GET BUFFER POINTER
000054 011646           MOV     @SP,-(SP)       ;... & BUILD BUFF END
000056 161016           SUB     @R0,@SP
000060 162016           SUB     (R0)+,@SP
000062 012627           MOV     (SP)+,(PC)+     ;SAVE RESULT ...
000064 000000 CR.UBF:   .WORD   0
                        .IFDF   BINARY          ;IN BINARY VERSION ...
                        MOVB    @R0,R1          ;... CHECK MODE
                        BICB    #376,R0
                        MOVB    R0,CR.ISW       ;IF ASCII CLEAR SWITCH
                        BEQ     CR.RDC          ;IF BINARY WANTED ...
                        MOV     (SP)+,R0        ;... SET PTRS & SWITCH
                        MOV     CR.IBP,R1
                        CLR     R2              ;SET INTERRUPT FLAG
                CR.BIN: CMP     R1,CR.IBF       ;INTERNAL BUFF EMPTY?
                        BNE     CR.RLP
                CR.ERD: MOV     CR.IBS,CR.IBP    IF SO RESET INTERNAL PTR
                        MOV     R0,-(SP)         SAVE USER BUFF PTR ...
                CR.RDC: .ENDC
000066 011627           MOV     @SP,(PC)+
000070 000000 CR.UBP:   .WORD   0
000072 005036           CLR     @(SP)+          ;ZERO UNDERWAY FLAG
000074 012737           MOV     #101,@#CR.CSR   ;ENABLE INT & GO FOR CD READ
       000101
       177160
000102 000207           RTS     PC              ;RETURN USER FOR NOW
```

```
                .IFDF   BINARY          ;WITH BINARY DATA ...
CR.RLP: SWAB    @R1                     ;... COMPLETE CONVERSION
        MOV     (R1)+,(R0)+             ;... & GIVE TO USER
        CMP     R0,CR.UBE               ;USER BUFFER FULL?
        BNE     CR.RIN                  ;IF NOT GET NEXT WORD
        MOV     R1,CR.IBP               ;OTHERWISE SAVE INT PTR
        TST     R2                      ;COME HERE ON INTERRUPT?
        BNE     CR.ODN                  ;IF SO MODE SW. SET
        MOV     @SP,-(SP)               ;ELSE MUST SIMULATE ...
        MOV     -(R2),2(SP)             ;... STORE PC & PS
        SUB     #16,SP                  ;... & DUMMY SAVE REGS.
CR.ODN: TST     (SP)+                   ;IGNORE RETURN PC
        BR      CR.DXT                  ;... & TAKE COMPLETION EXIT
        .ENDC


        ;INTERRUPT SERVICE ROUTINES:
        ;  A)  CHECK FOR ERROR & COLLECT INPUT:
000104 010046 CR.INT: MOV     R0,-(SP)          ;SAVE USER R0
000106 016700         MOV     CR.UBP,R0         ;GET USER BUFF PTR ...
       177756
000112 013746         MOV     @#CR.CSR,-(SP)    ;... & READER STATUS
       177160
000116 006326         ASL     (SP)+             ;CHECK FOR SPECIAL CASES
000120 103477         BCS     CR.ERR            ;GO RETRY IF ERROR
000122 100447         BMI     CR.DUN            ;CLEAN UP IF DONE
000124 011146         MOV     R1,-(SP)          ;NOW SAVE USER R1
                      .IFDF   BINARY            ;IN BINARY VERSION ...
                CR.ISW: BR      .+4             ;... USE APPROPRIATE CONVERSION
                      BR      CR.ASC
                ;  B)  BINARY CONVERT & STORE:
                      MOV     ...,R1            ;GET INT BUFF PTR
                      MOV     @#CR.DBR,-(SP)    ;... & INPUT
                      MOV     @SP,-(SP)         ;... 2 COPIES FOR LATER
                      COMB    1(R0)             ;ODD COLUMN?
                      BPL     CR.RST
                      ASL     @SP               ;IF SO SHIFT INPUT TO HIGH
                      ASL     @SP
                      ASL     @SP
                      ASL     @SP
                      CLRB    (R1)+             ;MAKE NXT INSTR = MOVB
                CR.RST: BISB    1(SP),-1(R1)    ;SET HIGH BYTE AS REQD.
                      MOVB    (SP)+,(R1)+       ;THEN LOW BYTE
                      MOV     R1,CR.IBP         ;SAVE PTR
                      CMP     (SP)+,#7400       ;NOW LOOK FOR EOF CARD
                      BNE     CR.RXT
                      DECB    @R0               ;... PUNCHED 12-11-0-1
                      BPL     CR.RXT            ;... IN CC 1 THRU 8
                      ASLB    @R0
                      BPL     CR.EOF            ;IF END IGNORE REST OF CARD
                      BR      CR.CXT            ;OTHERWISE TRY NEXT TIME
                      .ENDC
```

```
                        ;  C)  ASCII CONVERT & STORE:
000126 113701 CR.ASC:   MOVB    @#CR.DB2,R1     ;GET COMPACTED INPUT
       177164
000132 100002           BPL     .+6             ;CONVERT CODES >233 ...
000134 062701           ADD     #340,R1         ;... TO RANGE >140
       000340
000140 105710           TSTB    @R0             ;IF FIRST C.C. ...
000142 001011           BNE     CR.CVT
000144 020127           CMP     R1,#301         ;... LOOK FOR F.O.F.
       000301
000150 001472           BEQ     CR.EOF
                        .IFDF   BLANKS          ;FOR BLANK SUPPRESS ...
                        CMPB    R1,#337         ;... LOOK FOR SUPPRESS ON
                        BEQ     CR.ZON
                        .ENDC
                        .IFNDF  ONLY26&ONLY29   ;FOR DUAL PUNCH DRIVER ...
000152 020127           CMP     R1,#227         ;... CHECK IF 029 CTL
       000227
000156 001505           BEQ     CR.029
000160 020127           CMP     R1,#270         ;... OR 026 CTL
       000270
000164 001476           BEQ     CR.026
                        .ENDC
000166 010146 CR.CVT:   MOV     R1,-(SP)        ;CONVERT CARD CODE ...
000170 162701           SUB     #40,R1          ;FOR EACH 40 IN CODE ...
       000040
000174 100003           BMI     CR.STO          ;... ADD 21 & STRIP 43
000176 162716           SUB     #17,@SP         ;... TO GET TABLE INDEX
       000017
000202 000772           BR      CR.CVT+2
                CR.STO: .IFNDF  ONLY26&ONLY29
000204 066716           ADD     CR.TOS,@SP      ;PICK APPROP. TABLE
       177624
                        .ENDC
000210 060716           ADD     PC,@SP          ;COMPUTE ADDR OF BYTE READ
000212 162716           ADD     #CR.TBL-.,@SP
       000202
000216 113620           MOVB    @(SP)+,(R0)+    ;... & STORE IN BUFFER
000220 020067           CMP     R0,CR.UBF       ;BUFFER FULL?
       177640
000224 001452           BEQ     CR.EXT
000226 151710 CR.BXT:   BISB    @PC,@R0         ;IF NOT SET UNDERWAY FLAG
000230 010067 CR.CXT:   MOV     R0,CR.UBP       ;SAVE NEW POINTER
       177634
000234 012601           MOV     (SP)+,R1        ;RESTORE USER REGS,
000236 012600 CR.TXT:   MOV     (SP)+,R0
000240 000002           RTI                     ;... & EXIT
```

```
                        ;  D)  CARD COMPLETED:
000242 105037 CR.DUN:   CLRB    @#CR.CSR        ;STOP INTERRUP S
       177160
000246 105710           TSTB    @R0             ;IF NO PROCESS NG YET ...
000250 001427           BEQ     CR.RPT          ;... CONTINUE
000252 012600           MOV     (SP)+,R0        ;OTHERWISE RES ORE USER R0
000254 013746           MOV     @#CR.RSV,-(SP)  ;... & NOW SAVE ALL
       000044
000260 004536           JSR     R5,@(SP)+
000262 016700           MOV     CR.UBP,R0       ;SET USER BUFF PTR
       177602
                        .IFDF   BINARY          ;FOR BINARY VERSION ...
                        TSTB    CR.TSW          ;... CHECK IF BINARY READ
                        BNE     CR.BDN          ;IF SO ACTION ACCRODINGLY
                        .ENDC
000266 016701           MOV     CR.UBE,R1       ;FOR ASCII, SET END PTR
       177572
                        .IFDF   BLANKS          ;... & PERHAPS CHECK SUPPRESS
                CR.TSW: BR      .+4             ;SWITCH ON?
                        BR      CR.ADN          ;IF NOT NO SUPPRESSION
                        .IFNDF  MARKS
                        TSTB    @R0             ;TEST IF END OF FILE
                        BMI     CR.DXT          ;SKIP NEXT CALCULATION IF EOF
                        CMP     R0,R1           ;IF BUFFER FULL OMIT NEXT
                        BEQ     .+6
                        SUB     #8.,R0          ;OTHERWISE LOSE CC 73-80
                        .ENDC
                        CMPB    -(R0),#240      ;THEN TRAILING SPACES
                        BEQ     .-4
                        TSTB    (R0)+           ;ADJUST PTR WHEN DONE
                CR.ADN: .ENDC
000272 105041           CLRB    -(R1)           ;CLEAR REST OF BUFFER
000274 020100           CMP     R1,R0
000276 101375           BMI     .-4
000300 112721           MOVB    #215,(R1)+      ;MOVE IN CARRIAGE RETURN
       000215
000304 112721           MOVB    #012,(R1)+      ;MOVE IN LINE FEED
       000012
000310 016700 CR.DXT:   MOV     CR,R0           ;GET DDB ADDRESS
       177464
000314 000170           JMP     @14(R0)         ;TAKE COMPLETION EXIT
       000014
                        .IFDF   BINARY
                CR.BDN: MOV     CR.IBS,R1       ;FOR BINARY, INIT INT PTR
                        MOVB    @R0,R2          ;EXIT IF EOF SEEN
                        BMI     CR.DXT          ;ALSO SETS INTERRUPT FLAG
                        JSR     PC,CR.BIN       ;ELSE GO MOVE DATA TO USER
                        MOV     @#CR.SXT,R5     ;IF COME BACK, MORE READ REQD.
                        JMP     4(R5)           ;SO TAKE SYSTEM EXIT
                        .ENDC
```

```
                        CR.TBL:
                        ;PARITY ASCII CONVERSION TABLE FOR 029 PUNCH
                                .TENDF  ONLY26
000414    240                   .BYTE 240                 ;SPACE
000415    261                   .BYTE 261                 ;1
000416    262                   .BYTE 262                 ;2
000417    063                   .BYTE 63                  ;3
000420    264                   .BYTE 264                 ;4
000421    065                   .BYTE 65                  ;5
000422    066                   .BYTE 66                  ;6
000423    267                   .BYTE 267                 ;7
000424    270                   .BYTE 270                 ;8
000425    240                   .BYTE 240                 ;EMPTY
000426    072                   .BYTE 72                  ;:
000427    243                   .BYTE 243                 ;#
000430    300                   .BYTE 300                 ;@
000431    047                   .BYTE 47                  ;'
000432    275                   .BYTE 275                 ;=
000433    042                   .BYTE 42                  ;"
000434    071                   .BYTE 71                  ;9
                        ;
000435    060                   .BYTE 60                  ;0
000436    257                   .BYTE 257                 ;/
000437    123                   .BYTE 123                 ;S
000440    324                   .BYTE 324                 ;T
000441    125                   .BYTE 125                 ;U
000442    126                   .BYTE 126                 ;V
000443    327                   .BYTE 327                 ;W
000444    330                   .BYTE 330                 ;X
000445    131                   .BYTE 131                 ;Y
000446    240                   .BYTE 240                 ;EMPTY
000447    335                   .BYTE 335                 ;]
000450    254                   .BYTE 254                 ;,
000451    245                   .BYTE 245                 ;%
000452    137                   .BYTE 137                 ;_
000453    276                   .BYTE 276                 ;>
000454    077                   .BYTE 77                  ;?
000455    132                   .BYTE 132                 ;Z
                        ;
000456    055                   .BYTE 55                  ;-
000457    312                   .BYTE 312                 ;J
000460    113                   .BYTE 113                 ;K
000461    314                   .BYTE 314                 ;L
000462    115                   .BYTE 115                 ;M
000463    116                   .BYTE 116                 ;N
000464    317                   .BYTE 317                 ;O
000465    120                   .BYTE 120                 ;P
000466    321                   .BYTE 321                 ;Q
000467    240                   .BYTE 240                 ;EMPTY
000470    041                   .BYTE 41                  ;!
000471    044                   .BYTE 44                  ;$
000472    252                   .BYTE 252                 ;*
000473    251                   .BYTE 251                 ;)
000474    273                   .BYTE 273                 ;;
000475    134                   .BYTE 134                 ;\
000476    322                   .BYTE 322                 ;R
                        ;
```

```
000477  246             .BYTE 246               ;&
000500  101             .BYTE 101               ;A
000501  102             .BYTE 102               ;B
000502  303             .BYTE 303               ;C
000503  104             .BYTE 104               ;D
000504  305             .BYTE 305               ;E
000505  306             .BYTE 306               ;F
000506  107             .BYTE 107               ;G
000507  110             .BYTE 110               ;H
000510  240             .BYTE 240               ;EMPTY
000511  333             .BYTE 333               ;[
000512  056             .BYTE 56                ;.
000513  074             .BYTE 74                ;<
000514  050             .BYTE 50                ;(
000515  053             .BYTE 53                ;+
000516  336             .BYTE 336               ;^
000517  311             .BYTE 311               ;I
                        .ENDC

                ;PARITY ASCII CONVERSION TABLE FOR 026 PUNCH:
                        .IFNDF  ONLY29
000520  240             .BYTE 240               ;SPACE
000521  261             .BYTE 261               ;1
000522  262             .BYTE 262               ;2
000523  063             .BYTE 63                ;3
000524  264             .BYTE 264               ;4
000525  065             .BYTE 65                ;5
000526  066             .BYTE 66                ;6
000527  267             .BYTE 267               ;7
000530  270             .BYTE 270               ;8
000531  240             .BYTE 240               ;EMPTY
000532  137             .BYTE 137               ;0
000533  275             .BYTE 275               ;=
000534  340             .BYTE 340               ;@
000535  336             .BYTE 336               ;^
000536  047             .BYTE 47                ;'
000537  134             .BYTE 134               ;\
000540  071             .BYTE 71                ;9
                ;
000541  060             .BYTE 60                ;0
000542  257             .BYTE 257               ;/
000543  123             .BYTE 123               ;S
000544  324             .BYTE 324               ;T
000545  125             .BYTE 125               ;U
000546  126             .BYTE 126               ;V
000547  327             .BYTE 327               ;W
000550  330             .BYTE 330               ;X
000551  131             .BYTE 131               ;Y
000552  240             .BYTE 240               ;EMPTY
000553  273             .BYTE 273               ;[
000554  254             .BYTE 254               ;,
000555  050             .BYTE 50                ;(
000556  042             .BYTE 42                ;"
000557  243             .BYTE 243               ;#
000560  245             .BYTE 245               ;%
000561  132             .BYTE 132               ;Z
                ;
```

```
000562  055             .BYTE 55                ;-
000563  312             .BYTE 312               ;J
000564  113             .BYTE 113               ;K
000565  314             .BYTE 314               ;L
000566  115             .BYTE 115               ;M
000567  116             .BYTE 116               ;N
000570  317             .BYTE 317               ;O
000571  120             .BYTE 120               ;P
000572  321             .BYTE 321               ;Q
000573  240             .BYTE 240               ;EMPTY
000574  072             .BYTE 72                ;:
000575  044             .BYTE 44                ;'
000576  252             .BYTE 252               ;*
000577  333             .BYTE 333               ;[
000600  276             .BYTE 276               ;>
000601  246             .BYTE 246               ;&
000602  322             .BYTE 322               ;R
                ;
000603  053             .BYTE 53                ;+
000604  101             .BYTE 101               ;A
000605  102             .BYTE 102               ;B
000606  303             .BYTE 303               ;C
000607  104             .BYTE 104               ;D
000610  305             .BYTE 305               ;E
000611  306             .BYTE 306               ;F
000612  107             .BYTE 107               ;G
000613  110             .BYTE 110               ;H
000614  240             .BYTE 240               ;EMPTY
000615  077             .BYTE 77                ;?
000616  056             .BYTE 56                ;.
000617  251             .BYTE 251               ;)
000620  335             .BYTE 335               ;]
000621  074             .BYTE 74                ;<
000622  041             .BYTE 41                ;!
000623  311             .BYTE 311               ;T
                        .ENDC
```

```
                ;INTERNAL BUFFER FOR BINARY STORAGE:
                        .IFDF   BINARY
CR.BUF:         .IFNDF  MARKS
CR.BSZ=120.
                        .ENDC
                        .IFDF   MARKS
CR.BSZ=60.
                        .ENDC
                .=.+CR.BSZ
                        .ENDC
000001                  .END
```

```
000000 ERRORS


CR          000000RG    CR.AGN     000042R     CR.ASC     000126R
CR.BXT      000226R     CR.CSR  =  177160      CR.CVT     000166R
CR.CXT      000230R     CR.DB1  =  177162      CR.DB2  =  177164
CR.DUN      000242R     CR.DXT     000310R     CR.EOF     000336R
CR.ERR      000320R     CR.EXT     000352R     CR.INT     000104R
CR.IXT      000236R     CR.NAM     000014R     CR.NRY     000400R
CR.ONR      000016R     CR.OPN     000022R     CR.RPT     000330R
CR.RSV  =   000044      CR.STO     000204R     CR.SXT  =  000042
CR.TAL      000414R     CR.TFR     000042R     CR.TOS     000034R
CR.URE      000064R     CR.URP     000070R     CR.026     000362R
CR.029      000372R     PC      =%000007       R0      =%000000
R1      =%000001        R2      =%000002       R3      =%000003
R4      =%000004        R5      =%000005       SP      =%000006
.       =  000624R
```

APPENDIX A

PDP-11 CARD CODES

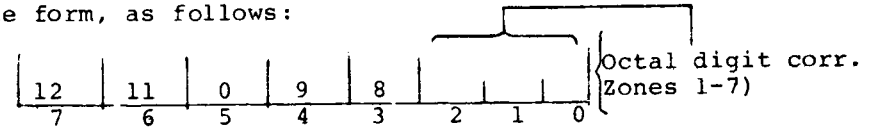
CHARACTER	ASCII	DEC029	DEC026	CHARACTER	ASCII	DEC029	DEC026
SPACE	24Ø	NONE	NONE	@	3ØØ	8 4	8 4
!	Ø41	11 8 2	12 8 7	A	1Ø1	12 1	12 1
"	Ø42	8 7	Ø 8 5	B	1Ø2	12 2	12 2
#	243	8 3	Ø 8 6	C	3Ø3	12 3	12 3
$	Ø44	11 8 3	11 8 3	D	1Ø4	12 4	12 4
%	245	Ø 8 4	Ø 8 7	E	3Ø5	12 5	12 5
&	246	12	11 8 7	F	3Ø6	12 6	12 6
'	Ø47	8 5	8 6	G	1Ø7	12 7	12 7
(	Ø5Ø	12 8 5	Ø 8 4	H	11Ø	12 8	12 8
)	251	11 8 5	12 8 4	I	311	12 9	12 9
*	252	11 8 4	11 8 4	J	312	11 1	11 1
+	Ø53	12 8 6	12	K	113	11 2	11 2
,	254	Ø 8 3	Ø 8 3	L	314	11 3	11 3
-	Ø55	11	11	M	115	11 4	11 4
.	Ø56	12 8 3	12 8 3	N	116	11 5	11 5
/	257	Ø 1	Ø 1	O	317	11 6	11 6
Ø	Ø6Ø	Ø	Ø	P	12Ø	11 7	11 7
1	261	1	1	Q	321	11 8	11 8
2	262	2	2	R	322	11 9	11 9
3	Ø63	3	3	S	123	Ø 2	Ø 2
4	264	4	4	T	324	Ø 3	Ø 3
5	Ø65	5	5	U	125	Ø 4	Ø 4
6	Ø66	6	6	V	126	Ø 5	Ø 5
7	267	7	7	W	327	Ø 6	Ø 6
8	27Ø	8	8	X	33Ø	Ø 7	Ø 7
9	Ø71	9	9	Y	131	Ø 8	Ø 8
:	Ø72	8 2	11 8 2	Z	132	Ø 9	Ø 9
;	273	11 8 6	Ø 8 2	[	333	12 8 2	11 8 5
<	Ø74	12 8 4	12 8 6	\	134	11 8 7	8 7
=	275	8 6	8 3	]	335	Ø 8 2	12 8 5
>	276	Ø 8 6	11 8 6	↑	336	12 8 7	8 5
?	Ø77	Ø 8 7	12 8 2	←	137	Ø 8 5	8 2

APPENDIX B

ALGORITHMS USED IN CR11/CM11 CARD READER DRIVER

B.1 HOLLERITH TO ASCII CONVERSION

Examination of the valid Hollerith character codes listed at Appendix A shows that in any one character there can be only one punch, if any, in control zones 12, 11, and 0. When translated by the CR11 Control into byte form, as follows:

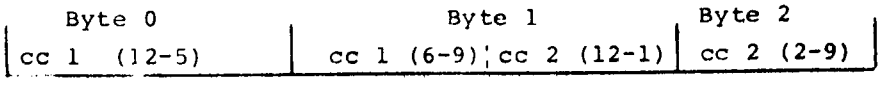


it can be seen that all characters must fall into one of the octal ranges: 0-37, 40-77; 100-137, 200-237. Moreover, within each range, the values are in fact restricted to the first seventeen. Basically, it is therefore possible to establish a table in four sections, each corresponding to one of these ranges, or two like tables if both 026 and 029 punches are considered.

Further, if the bytes so formed are transferred from the CR11 buffer into a register, values in the last range produce negative results by sign extension. If 340_8 is added to these, their range now becomes 140-177; a natural progression from the other three. Using a second register as a form of counter, a relative index to the required ASCII equivalent within its appropriate table section can be thus established simply by adding 21_8 to the low-order 5 bits of the Hollerith code for each time 40_8 can be successfully subtracted from the high-order three bits. (In practice, this is accomplished by subtracting 17_8 from one register containing the final index, to remove the 40 while adding 21, while reducing the counter register.) To the index must then be added the appropriate offset into the correct table for the punch concerned (0 for 029 and 104_8 for 026 if both tables are present). The address of the ASCII value required is merely the index added to a computed absolute table base.

B.2 BINARY PACKING

Basically, the packed format in which binary data is passed to a user program can first be considered as a problem of packing two 12-bit words representing card columns into three 8-bit bytes as:



or, in other words, the first column is shifted to the high-order position over the first two bytes while the second column remains in the low-order position in which it was read over the last two. Using a simple flip-flop type of switch, the algorithm distinguishes between columns 1 and 2 to accomplish this during the appropriate interrupts. Columns 3 and 4 require similar treatment. Owing to the byte addressing scheme of PDP-11, however, the result so obtain means that the bytes within each word are misplaced. A simple byte-swap when the word is passed to the user corrects this.

B.3 SWITCHING

The version of the driver which allows binary processing requires several switches as noted in the main text. Because the Monitor will never allow the driver to be called to perform more than one operation at a time, there is no need for the driver to be restricted to re-entrant code. As a result, the general form of switching used is of the form:

```
SWITCH:    BR  .+2
           BR  PROC.B
PROC.A:
```

When the low-order byte of SWITCH is cleared, the effective instruction at that point then becomes BR .+2 and the branch to process B is taken. If on the other hand that byte is then incremented, the instruction becomes BR .+4 and process A is entered.

A variation of this technique has been used at the start of the TRANSFER routine in the binary-type driver firstly to allow OPEN to stop a first read as described in the main text, and secondly to cause execution of the once-only code needed to initialize the internal buffer pointers. This extends the single fixed branch to a table:

```
CR.TFR:    BR  .+4       (Allows READ w/o to proceed)
           BR  CR.OXT    (Ignore first read after OPEN)
           BR  CR.ISP    (Initialize buffer pointers)
```

If OPEN is called, the switch byte is cleared causing the first transfer call to branch to CR.OXT (BR .+2). The routine at CR.OXT merely increments it back to the BR .+4 state and exits. The next entry at CR.TFR (or the first if OPEN is not called) takes branch to CR.ISP. The last instruction of this routine, which also immediately precedes CR.TFR, executes INCB @PC, hence finally setting the switch byte for

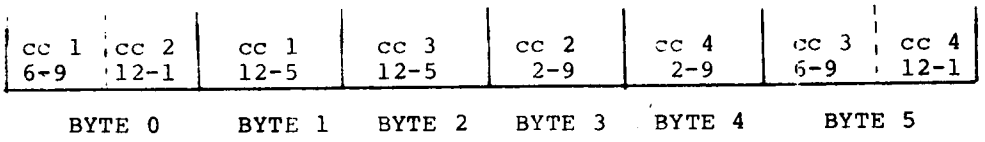
BR .+6 leading to all successive calls beginning normal execution immediately, until either the driver is re-initialized by a new OPEN or is removed from core and brought in afresh.

APPENDIX C

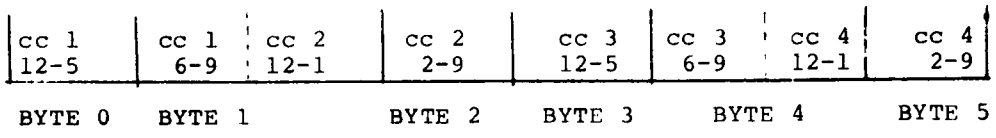
UNPACKING BINARY DATA FROM THE CR11/CM11 CARD READER DRIVER

A SUGGESTED ALGORITHM

1. Each four card-columns of data passed to a user program in binary format appear in memory as follows, when the byte-addressing scheme of PDP-11 is considered:



At first sight, a simple algorithm to restore the original 12-bit card column images from this format might seem a problem. If, however, the bytes in each word are first switched, the format now shows a more logical sequence:



and the solution reduces merely to one of splitting two like sets of three bytes into two 12-bit words. The first step, therefore, is to perform the necessary swap across all words transferred. Assuming the data has been read into a line buffer as defined under DOS, the appropriate code might be:

```
MOV     #LINE+4,RØ      ;GET BYTE COUNT FROM LINE HDR
MOV     (RØ)+,R1        ;... & BUMP POINTER TO FIRST DATA
ADD     RØ,R1           ;USE CNT TO SET LINE END
MOV     RØ,R2           ;SAVE START POINTER
SWAB    (R2)+           ;... & DO BYTE SWITCH
CMP     R2,R1
BLO     .-4
```

2. The column 1 image is simply obtained by taking byte 0 as the high-order part and byte 1 as the low-order part of a word which is then shifted until the required 12 high-order bits are right-justified. Column 2 image is similarly extracted from byte 1 and byte 2 except that no shift is needed. Using a simple flip-flop type switch to differentiate between odd and even columns, the necessary sequence might be:

```
        CLR     R2              ;INIT. FLIP-FLOP
A:      MOVB    (RØ)+,-(SP)     ;GET FIRST BYTE
        SWAB    @SP             ;... INTO HIGH ORDER SPOT
        MOVB    (RØ)+,@SP       ;... & 2ND INTO LOW
        COM     R2              ;FIRST COLUMN?
        BPL     B
        ASR     @SP             ;IF SO, RIGHT JUSTIFY
        ASR     @SP
        ASR     @SP
        ASR     @SP
        DEC     RØ              ;STAY AT SECOND BYTE
B:      BIC     #17ØØØØ,@SP     ;REMOVE GARBAGE
        .......                 ;REQD. IMAGE NOW ON STACK TOP
        .......                 ;... & CAN BE PROCESSED AS NECESSARY
        CMP     RØ,R1           ;END OF BUFFER?
        BLO     A               ;IF NOT, GET NEXT IMAGE
```

NOTES

a) The stack is used rather than a register to build the image, as this avoids the problem of possible sign extension in the operation at A+4. After processing, the image should of course be removed before proceeding to the next.

b) For mere storage of images in another buffer, a further register might be used as a pointer, e.g., MOV #BUFFER,R3 and all references to SP can then be changed to @R3 until the one at B, which should become (R3)+ to step to next word.

APPENDIX D

PREPARATION AND USAGE OF CR11/CM11 CARD READ DRIVER

D.1 PREPARATION

It has been shown that, by defining the relevant conditional parameters at assembly time, the user can tailor the card-reader driver to meet the particular needs of his installation. To allow him to do this, the driver is supplied as a source tape. The following paragraphs illustrate special points to be observed in preparing this tape for usage. It is assumed that the user is already familiar with the general operating procedures of the PAL-11R Assembler and Link-11 Linker.

D.1.1 Assembly

In order to enter the definitions for the required optional parameters, the user should specify that the keyboard will be used to supply input on Pass 1 only. Thus, assuming a disk-to-disk assembly with line-printer listings, the response to PAL-11R request for command input might be:

```
#CR.OBJ,LP:,LP:<KB:/PA:1,DF:CR.PAL
```

After this, the user can begin to type in the definitions he needs, e.g.:

```
BLANKS = Ø
BINARY = Ø
```

and then terminate by calling the Monitor to signal the end of this particular input as:

```
↑C              (CTRL/C
.END            (Two carriage-returns are essential after .END)
```

Thereafter, the assembly proceeds in the normal way.

For easy reference, the possible parameters and their effects are summarized below:

Parameter	Driver Version	Size
Nil	ASCII only - 026 or 029 (029 default)	200
DEFALT	ASCII only - 026 or 029 (026 default)	+1

Parameter	Driver Version	Size
ONLY26	ASCII only - 026 punch only	-51
ONLY29	ASCII only - 029 punch only	-51
BLANKS	Auto-deletion of cc 73-80 & trailing spaces by Special Card Control	+18
BINARY	Adds binary to other capabilities.	+161
MARKS	Restricted driver for 40-col CM11	0 on Basic -4 on BLANKS -30 on BINARY

NOTE

Signed size values should be added to the basic size for the ASCII-only version, e.g., driver assembled with ONLY29, BINARY, and BLANKS defined is 328 words long. Also first four mutually exclusive.)

D.1.2 Linking

The output file from PAL-11R should then be linked to paper tape, using Link-11. It must be origined at location Ø if it is to be used within DOS, i.e., the Command String in this case might be:

```
#PP:,LP:<CR/B:Ø/E
```

D.1.3 Inclusion in DOS Monitor Library

The DOS Monitor has already been set up to recognize the Card Reader Driver. Therefore, the user need merely incorporate the linked module into the Monitor Library, either by making the paper tape part of the input while building the system on disk by SYSLOD or by including it in the Monitor Library on a DECtape using MODS. The associated descriptions for these programs give details of method.

D.2 USAGE

In general, a card file consists merely of the data cards followed by an EOF card, then a blank. It has been shown that control cards may also be included to force the driver to function in a special way. The format for each of these is summarized below:

EOF	12-11-0-1	in c.c. 1 for ASCII, in c.c. 1-8 for Binary	
029 codes	12-0-7-9	in c.c. 1	If also in c.c. 80, card is symmetrical & can be used any way up.
026 codes	12-11-8-9	in c.c. 1	
Suppress Blanks	12-11-0-7-8-9	in c.c. 1	

After placing the card file in the hopper, the operator should ensure that power has been turned on and that both the MOTOR START & READ START buttons have been pressed (both indicators green). The read will then respond to any program request for input from device CR.

If an error occurs at any time, the Monitor message "AØØ2 12Ø6Ø" will indicate this. The operator should rectify the error if possible, replace the last card read with the remainder of the deck in the hopper, re-enable the reader and type CO to resume.

HOW TO OBTAIN SOFTWARE INFORMATION

Announcements for new and revised software, as well as programming notes, software problems, and documentation corrections are published by Software Information Service in the following newsletters.

Digital Software News for the PDP-8 & PDP-12
Digital Software News for the PDP-11
Digital Software News for the PDP-9/15 Family

These newsletters contain information applicable to software available from Digital's Program Library, Articles in Digital Software News update the cumulative Software Performance Summary which is contained in each basic kit of system software for new computers. To assure that the monthly Digital Software News is sent to the appropriate software contact at your installation, please check with the Software Specialist or Sales Engineer at your nearest Digital office.

Questions or problems concerning Digital's Software should be reported to the Software Specialist. In cases where no Software Specialist is available, please send a Software Performance Report form with details of the problem to:

Software Information Service
Digital Equipment Corporation
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

These forms which are provided in the software kit should be fully filled out and accompanied by teletype output as well as listings or tapes of the user program to facilitate a complete investigation. An answer will be sent to the individual and appropriate topics of general interest will be printed in the newsletter.

Orders for new and revised software and manuals, additional Software Performance Report forms, and software price lists should be directed to the nearest Digital Field office or representative. U.S.A. customers may order directly from the Program Library in Maynard. When ordering, include the code number and a brief description of the software requested.

Digital Equipment Computer Users Society (DECUS) maintains a user library and publishes a catalog of programs as well as the DECUSCOPE magazine for its members and non-members who request it. For further information please write to:

DECUS
Digital Equipment Corporation
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

READER'S COMMENTS

Digital Equipment Corporation maintains a continuous effort to improve the quality and usefulness of its publications. To do this effectively we need user feedback -- your critical evaluation of this manual.

Please comment on this manual's completeness, accuracy. organization, usability and readability.

Did you find errors in this manual? If so, specify by page.

How can this manual be improved?

Other comments?

Please state your position. _____ Date: _____

Name: _____ Organization: _____

Street: _____ Department: _____

City: _____ State: _____ Zip or Country _____

— — — — — — — — — — — — — — — — Fold Here — — — — — — — — — — — — — — — — — —

— — — — — — — — — — — — Do Not Tear - Fold Here and Staple — — — — — — — — — — —

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

digital

Digital Equipment Corporation
Software Information Services
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754